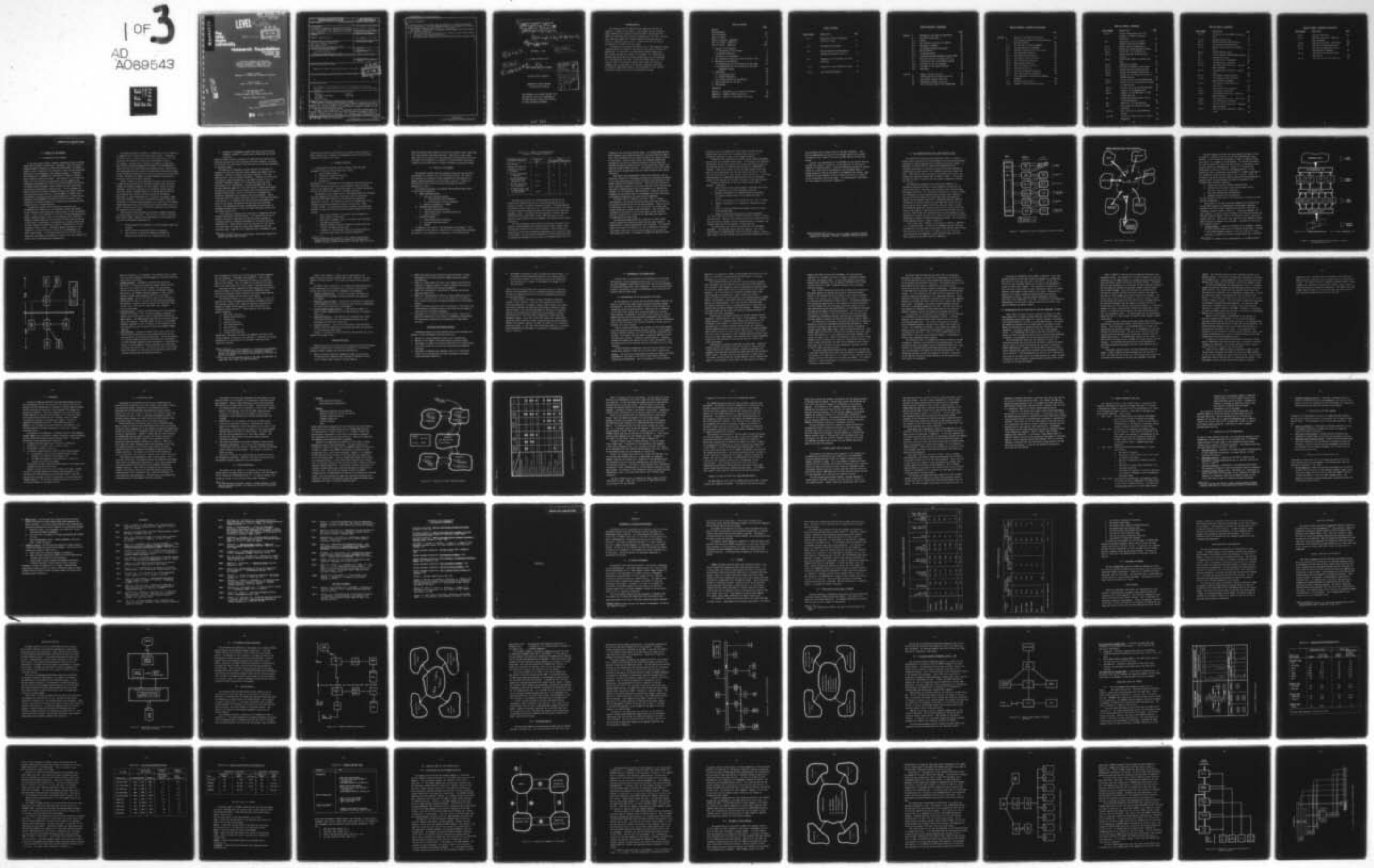
AD-A069 543   OHIO STATE UNIV RESEARCH FOUNDATION COLUMBUS   F/G 9/4
A SIMULATION LANGUAGE FOR EVALUATING INFORMATION PROCESSING SYS--ETC(U)
SEP 78   T G DELUTIS   DAAG29-77-G-0203

UNCLASSIFIED   ARO-15356.1-A-M   NL

1 OF 3
AD
A069543



ARO 15356.1-A-M

RF Project 760792/710632
Final Report

LEVEL II

12 SC

the
ohio
state
university

research foundation
1314 kinnear road
columbus, ohio
43212

A SIMULATION LANGUAGE FOR EVALUATING INFORMATION PROCESSING SYSTEMS FROM A DBMS AND USER PERSPECTIVE: EXTENSIONS TO THE INFORMATION PROCESSING SYSTEM SIMULATOR

Thomas G. DeLutis
Department of Computer and Information Sciences

For the Period
August 1, 1977 - January 31, 1978

U.S. ARMY RESEARCH OFFICE
P.O. Box 12211
Research Triangle Park, North Carolina 27709

Grant No. DAAG29-77-G-0203

March, 1979

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER

2. GOVT ACCESSION NO.

3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle)
A Simulation Language for Evaluating Information Processing Systems from a DBMS and User Perspective: Extensions to the Information Processing System Simulator

5. TYPE OF REPORT & PERIOD COVERED
Final; 8/1/77 - 1/31/78

6. PERFORMING ORG. REPORT NUMBER
OSURF Project #710632

7. AUTHOR(s)
Thomas G. DeLutis, Ph.D.

8. CONTRACT OR GRANT NUMBER(s)
Grant #DAAG 29 77 G 0203 new

9. PERFORMING ORGANIZATION NAME AND ADDRESS
Ohio State Research Foundation
200 Research Center
1314 Kinnear Rd., Columbus, Ohio 43212

10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS

11. CONTROLLING OFFICE NAME AND ADDRESS
U. S. Army Research Office
Post Office Box 12211
Research Triangle Park, NC 27709

12. REPORT DATE
September 1, 1978

13. NUMBER OF PAGES

14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)

15. SECURITY CLASS. (of this report)
Unclassified

15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
NA

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

DDC
RECEIVED
JUN 7 1979
C

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

NA

18. SUPPLEMENTARY NOTES

The findings in this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Modeling
Performance Evaluation
Data Base
Computer System
On-line
Real-time

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The process of analyzing and designing contemporary information systems is a complex one. Models of proposed system designs can be extremely helpful in determining their behavior prior to the expenditure of manpower, dollars and time on unsatisfactorily performing systems.

The goal of this research was to investigate simulation methodologies and language capabilities which could ease model synthesis and analysis efforts in support of system analysis and design. This research focused specifically on data base subsystems and on satisfaction of user performance requirements. (over)

next page

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20 (continued)

An integral part of the research was the modeling of a complex information system architecture to test the appropriateness of the information processing system simulator design facility (IPSS) as a design tool.

This document describes the investigative and modeling efforts and reports on the following conclusions:

1. IPSS is a useful tool for modeling in a topdown, modular manner complex information system architectures, and
2. The proposed new capabilities would significantly reduce modeling effort.

(6) A SIMULATION LANGUAGE FOR EVALUATING INFORMATION PROCESSING SYSTEMS FROM A DBMS AND USER PERSPECTIVE: EXTENSIONS TO THE INFORMATION PROCESSING SYSTEM SIMULATOR.

(9) FINAL rept. 1 Aug 77-31 Jan 78,

(10) THOMAS G. DELUTIS, ~~PH.D.~~

(11) 1 Sep 78

SEPTEMBER 1, 1978

(12) 270 p.

U. S. ARMY RESEARCH OFFICE

(18) ARO

(19) 15356.1-A-M

(15) GRANT NUMBER DAAG 29-77-G-0203

THE OHIO STATE UNIVERSITY

APPROVED FOR PUBLIC RELEASE;
DISTRIBUTION UNLIMITED.

Accession For	
NTIS GRA&I	✓
DDC TAB	
Unannounced	
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or special
A	

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION, UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

79 06 05 031

267 360

ACKNOWLEDGEMENTS

This research activity represents the cummulative efforts and support of a number of individuals. John Chandler and Joe Brownsmith spent countless hours analyzing SIDPERS and IDMS, synthesizing IPSS models and identifying and correcting program bugs in the IPSS. Without their dedicated support this research would not have been possible. Finally, a heartfelt thank you to Dr. Lawrence Rose for the hours he has devoted to the editing of this report.

The support and assistance provided by the U.S. Army Computer Systems Command personnel in support of the modeling experiments was especially appreciated. Mr. Alan Curry provided overall project guidance and Walt Kwiatek provided the liaison to the Command. Mr. Brownsmith and Chandler have asked that the following individuals also be recognized: Mr. Frank Hunter for assistance on IDMS and SIDPERS details; Mr. Milt Schul for PDP-11/70 assistance; Captain Jerry Jones for additional computer time; the personnel of the Systems Integration Directorate for technical operational support; and the personnel of the Technical Training and library Division for providing office space.

TABLE OF CONTENTS

TABLE OF FIGURES

TABLE OF CONTENTS: APPENDICES

TABLE OF CONTENTS: APPENDICES (continued)

TABLE OF FIGURES: APPENDICES

TABLE OF TABLES: APPENDICES

TABLE OF TABLES: APPENDICES (continued)

1.0 SNYOPSIS OF THE RESEARCH

1.1 INTRODUCTION TO THE PROBLEM

In the past quarter century, computer technology has had a profound impact upon the manner in which data is processed. With each advance in computer hardware technology has come associated advances in software capability both in respect to the processing of data into more meaningful packets and in the management of the computer hardware. A seemingly never ending cyclic phenomenon has developed with regard to computer based information processing systems. Each expansion in the processing capabilities only increases the appetite of the user community and consequently new demands are placed on the system designers and analysts to increase system throughput, reduce response time and provide more services while maintaining or reducing costs. This has mandated the better utilization of expensive computer hardware and software resources and associated personnel and thus, the development of complex system architectures incorporating sophisticated supervisors to handle the job scheduling, task management and resource allocation functions, multi-processors to increase throughput and integrated distributed data bases to meet the user's information needs.

Contemporary computer based information processing system architectures are reflective of the information systems they support. As the information needs become more sophisticated, so does the computer system. In part, this phenomenon is the result of new technologies which permit increased volumes of data to be incorporated into information systems at reduced operating cost and faster retrieval time per quantum of data stored. Unfortunately, the complexities of modern society have not only increased the enterprise's need for information but also the transiency of data's value. Furthermore, needed information is more integrative in nature in order to reflect the interdependencies of modern organizations. Ill-designed systems which do not meet their operational requirements are expensive in terms of the time, manpower and dollars lost in their design and implementation.

The design and analysis problem is a complex one, and its complexity is augmented by the costs associated with such systems. Furthermore, the analyst cannot, in most cases, rely on past experiences since each system must be tailored to the particular environment in which it will operate. The need exists, therefore, for modeling technologies which provide the analyst/designer with insights into the behavior of complex on-line, real-time information processing system. Additionally, such technologies must be attuned to the descriptive and evaluative needs of the information processing system so as to minimize the cost and effort for their use both with regard to synthesizing models and evaluating statistics resulting from their execution.

Ideally, models provide the designer and analyst with the ability to identify and characterize an information system's user, data base, software and hardware components and their logical, control and data interfaces. During the lifecycle of an information system, modeling activities provide a feedback capability to the designer and analyst which enhances their ability to attain and maintain performance objectives. By being representations of systems or subsystems, models serve three purposes -- as an aid to design by providing insight into the behavior of selected system components; as a predictor of performance changes that would result for alterations to existing systems; and as a guide to monitoring activities by identifying components to be monitored and statistics to be collected.

Discrete event simulation models can provide a number of benefits to system designers and analysts. They can also be expensive endeavors which remain unused. Among the important benefits to be achieved from this form of performance assessment are:

1. Insight gained into the behavior of the constituents comprising a system,
2. Identification of performance sensitive components,
3. Identification of system and component performance in response to user activity (e.g., volume and mix), and

4. Estimates of performance changes which would result from new hardware, software[1], and data base architectures and system procedures.

Simulation models serve as predictors of system activity; they are cost effective only when the costs associated with model synthesis, validation and evaluation are significantly lower than the cost of building benchmarks or prototype systems.

The modeling of contemporary systems requires the inclusion of submodels for data base, data base access software and support facility components. Without a systematic and modular approach to the total modeling effort, costs can become prohibitive. Therefore, the first step in achieving a modeling capability is the identification of a set of functional components common to the information processing system architectures to be investigated. A second requirement for a cost effective modeling capability is the availability of models which permit the analyses of alternative architectures and/or system activity with a minimum of modification. Third is the ability to incorporate new evaluative technologies and procedures into existing models. Fourth, simulation models must produce meaningful statistics from model evaluation; in particular, it is important that the statistics they generate are useful to the analyst and designer and that they can be validated. Finally, the performance assessment system should allow the reuse of component characterizations in order to effect technology transfer and thus reduce overall cost.

To be effective, simulation facilities must be compatible with the designer and analyst's evaluation goals. This implies the ability to perform a spectrum of modeling tasks from macroscopic analyses of preliminary designs to detailed assessments of instruction execution characteristics. Model synthesis must be consistent with the information known about the system and output data commensurate with the evaluation needs. Required is an integrated view of information system

[1] Software includes operating system modules, application programs and general data base access software.

activities and interfaces which permits a top down, modular approach to model synthesis and is capable of accommodating present and anticipated hardware and software architectures.

1.2 RESEARCH OBJECTIVES

The Information Processing System Simulator (IPSS) has been developed[(2)] with four general objectives in mind:

1. Ease of model synthesis,
2. Modularity in model construction,
3. System self-containment, and
4. Portability.

Additionally, it is recognized that the system must maintain executional efficiency and must be usable by a wide spectrum of users, from the novice to the sophisticate. A continuing goal is to further the development of IPSS within the framework of the above objectives.

The purpose of this research was to investigate methods for extending the IPSS language and statistical facilities, focusing upon those facilities which could increase the definitional and evaluative capabilities of IPSS. The research has led to the incorporation of new language and statistics gathering features into the IPSS. These include (a) DBMS -- associated definitional statements and performance measures, and (b) user and task-oriented evaluative capabilities which provide:

1. Definitional statements to ascertain the performance of a model relative to user goals;
2. Definitional statements specific to network and hierarahical type DBMS architectures;
3. Enhanced standard statistical outputs to provide additional insights into model behavior; and
4. Interfaces to internal model facilities thereby permitting additional model control and statistics generation.

---

(2) IPSS's development was sponsored in part through research grants GN-36622 and SIS-75-21648 from the National Science Foundation. (Office of Science Information Services) to The Ohio State University.

These new features permit modeling activities to be more closely identified with actual analysis and design functions. Additionally, they reduce model antificiality and provide more meaningful and identifiable statistics. The benefit to the modeler is a reduction in time and effort required to synthesize and validate models of complex information processing systems.

1.3 RESULTS OF THE RESEARCH

The research proposal identified modeling constructs and statistics as candiates for inclusion into the IPSS design facility. The vehicle for assessing the usefulness of these extensions was the modeling of a complex system architecture to be described in Section 3. Details of the evaluative efforts and the results established are found in Appendices A, B and C.

For evaluative purposes, the proposed IPSS extensions were placed in the following categories:

A. MODEL STRUCTURE RELATED
   1. Data Base Subsystem Submodels
      a. IPSS Data Base Access Component
      b. IPSS Data Base Structure Component
   2. Newwork of Submodels as a Model

B. LANGUAGE EXTENSION RELATED
   1. Data Base Subsystem Definition
   2. System Effectiveness Evaluation Definition
   3. Network Definition

C. STATISTICS RELATED
   1. System Effectiveness Evaluation
   2. Data Base Subsystem Behavior
   3. General

In Appendix C the results of the evaluation are discussed. In all 37 statements and 123 statistics identified in the proposal were evaluated. The results of the evaluation are summarized in the following table.

Table 1.3-1. Summary of the Evaluation of Proposed IPSS Language Extensions

ASSESSMENT CONCLUSION	DISPOSITION CODE	TOTAL	
		Statements	Statistics
Not Examined	NE	6	20
Examined and Rejected	ER	0	0
Examined and Found to be Useful -			
Has been implemented as proposed	EU-I	16	0
Has been implemented but merged into another facility	EU-IM	6	7
Has been defined and not implemented	EU-D	0	93
Has been defined, used in paper models, but not implemented	EU-DP	9	0

The modeling activities also demonstrated the feasibility and suitability of applying IPSS and its underlying methodology to the synthesis of models of complex information systems. The system modeled to demonstrate these capabilities was a host/back-end processor configuration which supported interactive application processing and a data base management system. The modeled software processes included a detailed characterization of application loading, DBMS processing, and the operating system function of task management, job scheduling and resource allocation.

The IPSS methodology proved capable of providing the modeler with a modular, top-down approach to system analysis and model synthesis thus facilitating a solution to this complex problem. Several models were constructed, each of which reflected system details of a particular aspect of the system. The ease of model expandability in both breadth and depth was demonstrated through the construction of independent models and their

integration into an overall complete model. The SIDPERS/IDMS workload model represents the software processes of the applications and IDMS in detail while characterizing the operating systems and computer hardware as black boxes. From this basic model, two subsequent models were developed by splitting the software processes at the host/back-end interface and adding characterizations of the appropriate hardware and operating system software functions. The synthesis of the final overall model was easily accomplished since the system interfaces were clearly identified.

These models were independently tested, each producing a set of statistics reflective of the modeled processes and computing environment. In all cases queueing and utilization statistics were obtained for the modeled software and hardware resources. These statistics provide an easily understood synopsis of activities since one or more IPSS services were used to represent major units of processing (e.g., application programs, TP line scheduling, IDMS processing). Statistics were also automatically collected on each of the hardware facilities indicating both potential bottlenecks and under-utilized components.

The models were all internally verified. That is, through experimentation, the internal processing consistency was verified to be accurately reflective of the real world processes. Due to the lack of operational data, however, it was not possible to validate them through comparison to actual system performance. Thus, although some confidence can be placed in the relative values of the statistics reported in Appendix B no validation has been made with respect to the magnitude of the data values.

The experiment demonstrated the suitability of IPSS to model complex systems while requiring minimal time for model synthesis. The modeling effort required approximately four man months. The resulting models have proven to be versatile as well as adaptable to changing requirements indicating that significant portions can be used for other application areas. For example, the DMBS and operating system

processes as well as hardware facilities need not be modified when another major application area is modeled, thus resulting in a substantial reduction in future related modeling efforts.

In addition, this experiment has served to focus future development work as well as to validate current goals. In particular, the experiment demonstrated the need for simulation facilities specifically attuned to the characterization of integrated data structures as represented in IPSS/DBS. These initial models did not use these features since they were not fully operational, and thus the logical structure of the data base was characterized at a more abstract level than desired. Furthermore, experimentation with IDMS record mapping policies was not attempted because of the complexities involved in relation to the time available.

A number of experiments on the basic models became readily apparent. These were:

1. Identification of the maximum number of terminals which could be supported as the system approaches saturation;
2. Response time behavior for system processes and resources as loading increases and/or mix of transaction types is varied;
3. Effect on throughput as the transaction error rate is varied;
4. Response time behavior as a function of transaction arrival rate;
5. Effect on system throughput and response time with a m-way, multi-threaded DBMS;
6. Impact of presorting transactions on system throughput.

The initial model did not contain operating system or computer hardware characteristics. However, later models did have these characterizations. Thus the effect of changes in their characterizations on throughput and response time could also be examined in future analyses.

The modeling was done using the original IPSS, now designated IPSS/BASIC. The IPSS/DBS features were not employed. As a result detailed models of DBMS behavior could not be easily modeled. However, paper models using the proposed syntax were written comparatively quickly

(one man-month which included review of the IDMS literature). This exercise demonstrated the value of the IPSS/DBS capabilities to reduce the modeling effort needed to characterize DBMS software and data base scheme.

Finally, the incorporation of the IPSS/DBS into the current IPSS has extended the concept of information systems modeling into the area of networks of simulation models. Since the IPSS/DBS was designed to execute either as a stand alone or asynchronously with IPSS/BASIC models, it became necessary to devise a mechanism to accommodate this concept. This mechanism has been extended so that an arbitrary collection of nodes[3] can be incorporated into an IPSS model. The implementation requirements for this concept are discussed in Appendix C.

---

[3] An IPSS model node is defined to be an uniquely identified submodel comprised of IPSS/BASIC, IPSS/DBS or IPSS/BASIC-IPSS/DBS components.

2.0 THE INFORMATION PROCESSING SYSTEM SIMULATOR (IPSS)

The Information Processing System Simulator (IPSS) is a special purpose digital simulation facility designed to facilitate the performance assessment of complex computer based information processing systems. It represents the realization of a generalized methodology for the performance assessment of information systems and has been designed to be applicable to every phase of the information processing system life cycle. Additionally, models developed for one phase can be easily and inexpensively modified in both scope and depth of detail. This enables the model to evolve in parallel with the system's progression through its lifecycle.

The methodology employed by IPSS divides the elements of an information processing system into three categories according to the function they serve: data bases, services which access the data bases, and support facilities. As illustrated in Figure 2-1, a system is viewed as a hierarchy of services and data bases with each level supporting the service and data needs of the next higher level and requesting support from the adjacent lower level. Performance measures identify the behavior of services relative to their data base access activity, acquisition and use of support facilities, and use by other services. The methodology emphasizes data base access activity and has identified those system elements at each data base level which contribute to this activity. Within IPSS, the characterization of system elements has been formulated in a manner such that their interaction during model evaluation automatically produces the desired performance measures. An illustration of the modeling process is shown in Figure 2-2.

IPSS is more than a language in the sense of a GPSS or a SIMSCRIPT; it is a complete system. In addition to statements defining resources and tasks for the system under investigation, IPSS contains special statements directing the IPSS system in a number of auxiliary functions including the use and management of user defined library facilities, model compilation, and model execution sequencing.

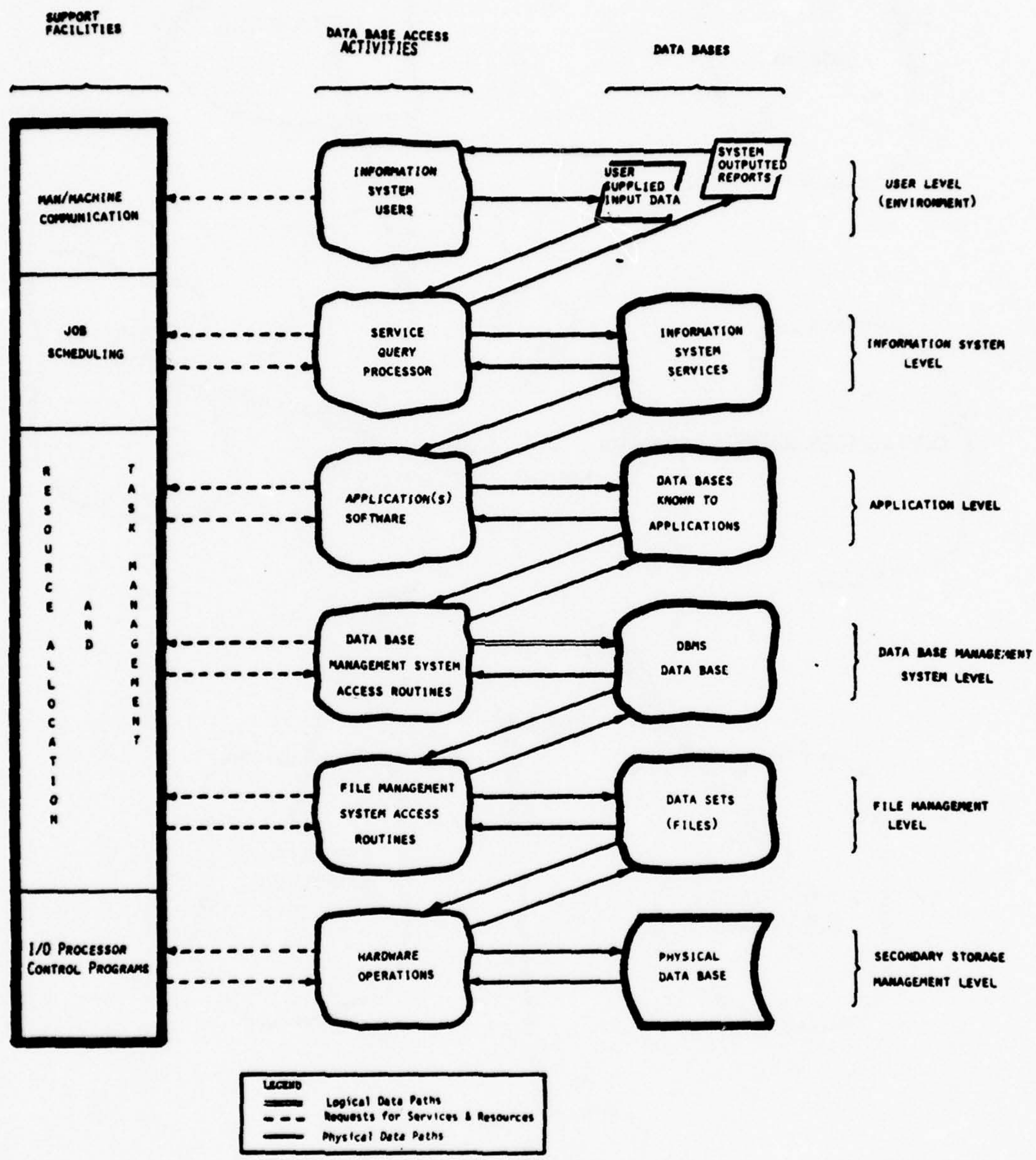


Figure 2-1 Hierarchical View of Information Processing Systems

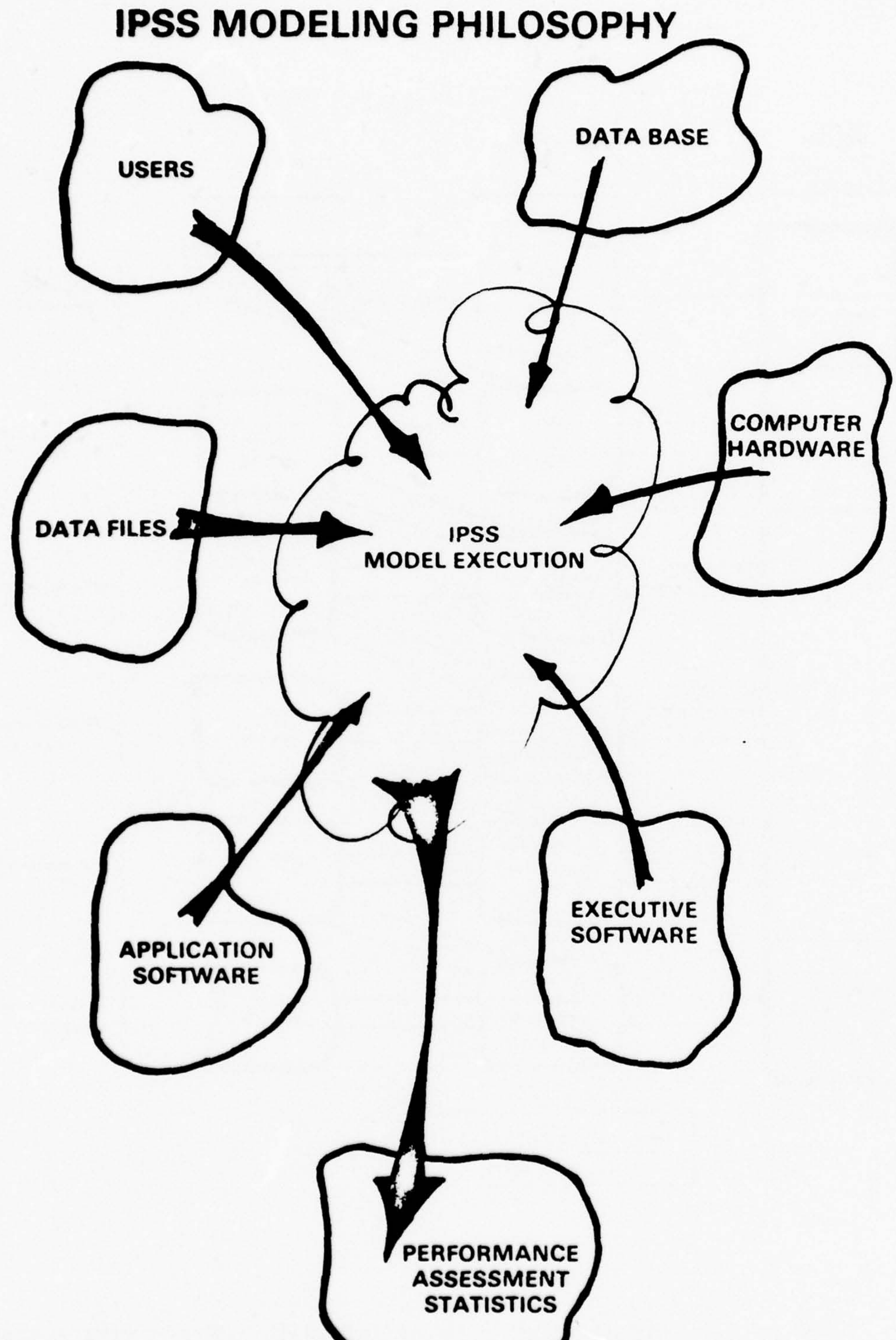


Figure 2-2 IPSS Modeling Philosophy

The IPSS language facilities have been carefully designed to focus the modeler's synthesis activities on the IPSS methodological underpinnings. In this manner the development of models of complex systems can be done in a systematic, top down manner. This affords the modeler the same advantages that these techniques provide in information system analysis, design and implementation. Figure 2-3 illustrates the three phases in the use of IPSS for IPSS model synthesis. At the begining all that is available to the modeler[4] is general knowledge of the information system (at some stage in its life cycle). Phase I is an implicit one during which the modeler employs the IPSS methodology to separate overall system knowledge into specific knowledge of the following four functional components:

A. System users and their request characteristics;

B. Services to be provided to users and to other services, and their inter service connections;

C. Data base resources and configurations;

D. Hardware resources and configurations.

The IPSS language has been specifically formulated to characterize these functional components.

During Phase II the above functional components are characterized via the IPSS language into independently defined model components. Presently five distinct model components are possible to describe an information system as shown in Figure 2-4. (The sixth component, the model director, is used to control simulation activities during Phase III, the model execution phase.) Not all are required to model a particular computer facility. Additionally, more than one of each component (except model director) can appear in an IPSS model. A synopsis of the role of each component follows:

1. SYSTEM RESOURCES -- contains definitions for all hardware, software and data resources comprising an information system model. Included in the SYSTEM RESOURCES component is the IPSS supplied clockwork mechanism to schedule and control simulated events and to determine

[4] A modeler is assumed to be a systems/analyst or a systems designer.

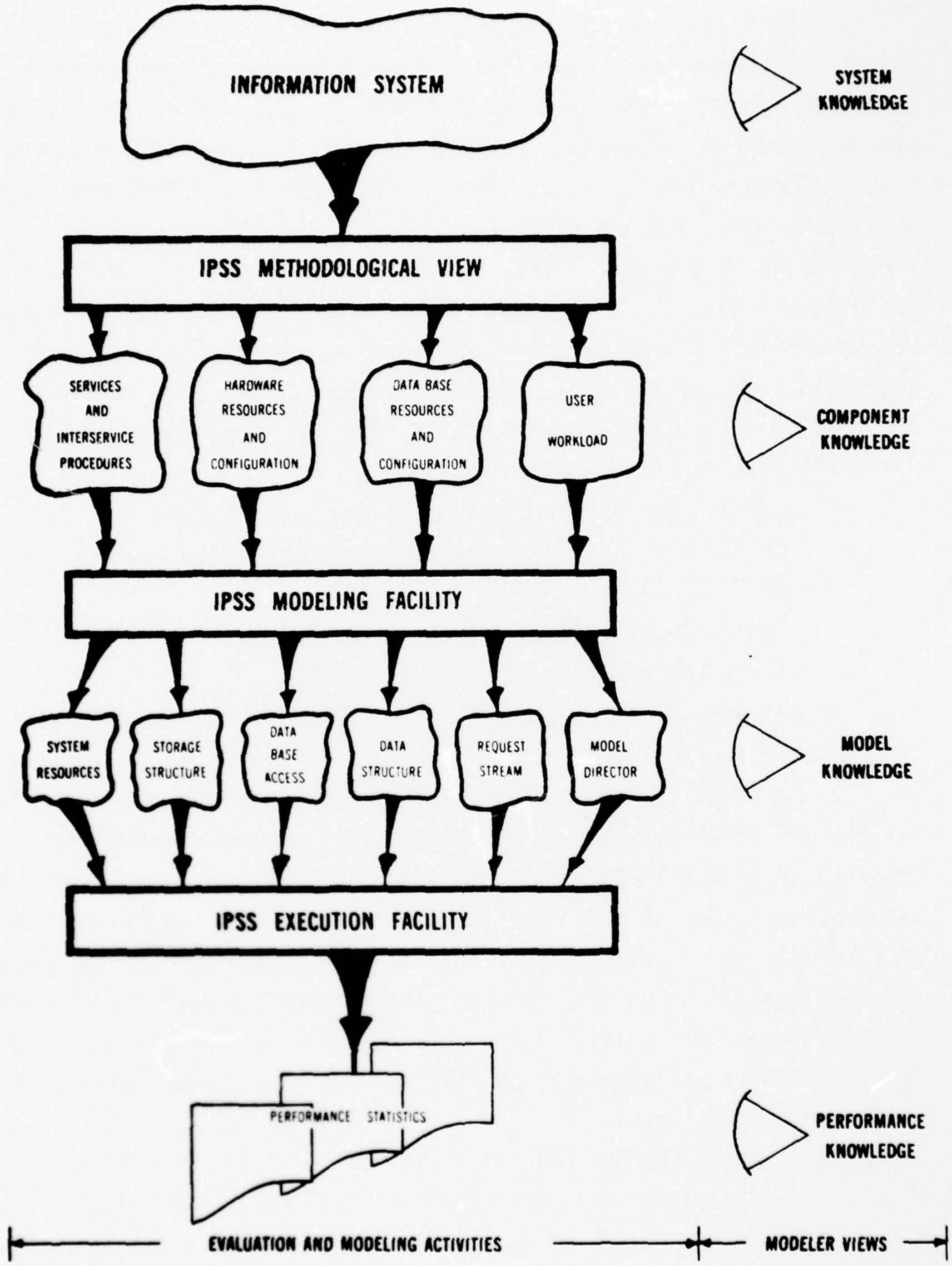


Figure 2-3 Relation Between IPSS Methodology, Language and Execution Facility

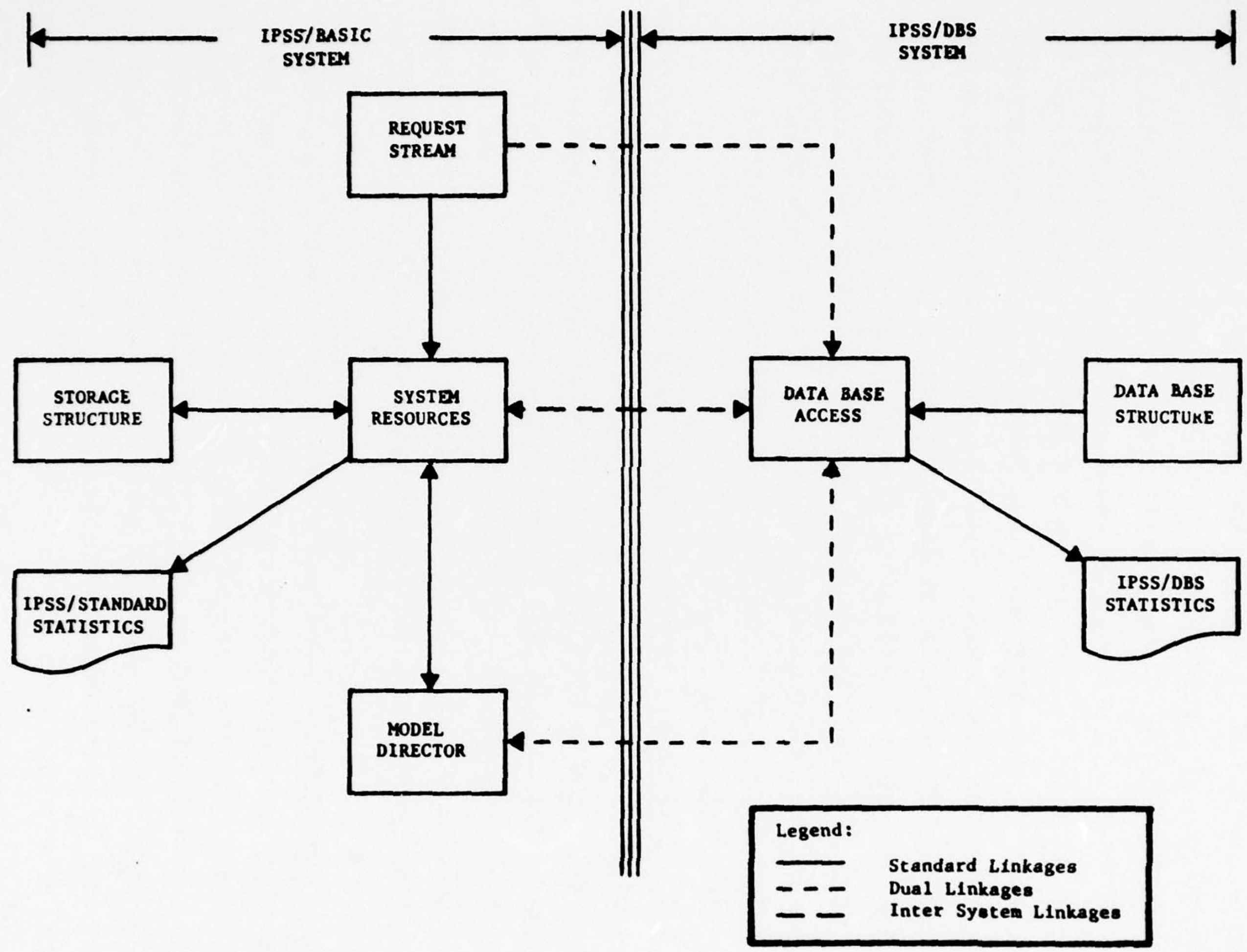


Figure 2-4 Components of an IPSS Model (with DBS Extension)

when the simulation is to terminate. The clockwork logic is based on the next most immediate event philosophy for controlling discrete event digital simulations;

2. STORAGE STRUCTURE -- supplements the definition of data set facilities. It provides the model with general definitions for data set files in terms of their formats, size and location relative to each other and their location in secondary storage. Additionally, the STORAGE STRUCTURE component is used to define secondary storage space management policies for the model;
3. REQUEST STREAM -- defines the external requests for information system services; that is, the users of the system being modeled. The modeler is required to define types of users and their inter-arrival times. IPSS converts these times into a composite arrival time stream;
4. DATA BASE ACCESS -- contains the definitions of all the resources required by the DBMS. These include the hardware resources of buffers and user work areas as well as application programs and DBMS software. All DBMS-related entity type facilities are defined within the component. DATA BASE ACCESS is similar to the SYSTEM RESOURCES component in that it contains its own simulation clockwork mechanism similar in purpose to that of the REQUEST STREAM component.
5. DATA STRUCTURE -- provides the modeler with a set of facilities which allows the definition of logical data structures and the characterization of relationships among them. This can be applied to a variety of DBMS architectures and application environments. The DATA STRUCTURE component permits the modeler to investigate the effects on system behavior caused by alternate set, record type, and access path definitions. The definitional facilities provided enable the modeler to investigate a wide spectrum of logical data structure organizations and allocation policies.

Also illustrated in Figure 2-4 is the division of the IPSS components into two subsystems -- IPSS/BASIC and IPSS/DBS[5]. This has been done to permit a modeler to concentrate his efforts on a particular aspect of the problem. Thus, the capability to model only the data base or the system aspects of a total system is provided the modeler.

Associated with each component are IPSS language facilities which permit the modeler to define and characterize information system entities and attributes, gather statistics and control model behavior. Over 100 such statements are available to the modeler.

The function of Phase III is to assemble the model, initiate its execution and display the generated performance assessment statistics. IPSS provides a modeler with a number of statistics concerning the behavior of modeler defined entities and IPSS supplied built-in information system services.[6] These statistics fall into eight general categories:

1. Operational Statistics,
2. Request Stream Statistics,
3. I/O Activity,
4. Queueing Statistics,
5. Utilization Statistics,
6. Wait Statistics,
7. Service Statistics, and
8. Task/Activity Statistics.

Additionally, the modeler can employ the complete facilities of the Fortran language to develop his own statistics. Statistics are printed automatically at the conclusion of each model simulation unless explicitly inhibited.

---

[5] This configuration of IPSS components is currently being implemented. The CSC model reported here employed only IPSS/BASIC components. However, the IDMS was also written using IPSS/DBS facilities but has yet to be validated.

[6] These services are characterizations of the basic I/O operations for tape, DASD, unit record, and terminal devices.

Finally, IPSS contains a library facility which permits the modeler to retain previously defined models, model components and code segments for later referral. This feature can greatly reduce modeling time.

An implicit goal of the IPSS is to achieve the maximum applicability of the methodology and the IPSS facility. Therefore, the following criteria have served as guidelines throughout its development.

1. Breadth of Applicability -- the ability to model the behavior of contemporary and foreseeable system architectures and operating environment;
2. Functional View of Systems -- the ability to identify and characterize system components and activities based on their function independently of a particular architecture or environment;
3. Top Down, Modular Model Synthesis -- the ability to model to a level of detail commensurate with the desired analysis of research objectives;
4. Extensible Structure -- the capability to incorporate new, higher level descriptive facilities and performance measures into the methodology and simulation system; and
5. Flexibility of Use -- the ability to be used by a wide spectrum of modelers from the experienced system analyst/designer and researcher to the practitioner and student.

Examples illustrating the broad flexibility and applicability of the extended IPSS system follow.

Modeling Activities

Modeling activities are those activities concerned with the development of evaluative capabilities for examining the behavior of existing and proposed software, hardware and data base architectures.

1. Models of specific data base management systems for the purpose of examining their behavior under a variety of different criteria such as loading, data structure, and data mapping.

2. Models for evaluating the influence of data structures on system performance over a broad spectrum of DBMS loadings and support architectures.
3. Models of systems where the central site computing facilities are comprised of computer networks in order to examine the effects of alternative scheduling and tasking philosophies.
4. Models for evaluating the interaction of front-end and back-end computing capabilities.
5. Models for investigating the effects of queue scheduling policies for basic system activities such as job scheduling, task management, resource allocation, and paging strategies for virtual memory systems.
6. Models to evaluate data base security procedures; in particular, the identification of authorized users to the system and the impact of such activities on overall performance.
7. Development of a simulation capability and support library which permits the modeling of a wide range of contemporary architectures for the purpose of evaluating new technologies as they become available.

Performance Measurement Research

Performance measures are those measures which assist designers and analysts in their performance evaluation activities.

1. Measures of system performance based on user oriented goals.
2. Measures of the impact of new applications on current systems.
3. Development of higher level measures which permit the identification of critical system activities based on applications performance objectives.
4. Development of measures and procedures which aid in identifying (from a set of alternatives) the "best" alternative to achieve system performance objectives.

5. Development of measures to assist the data base administrator. (It is assumed that data base administration is an on-going activity and its major function is to maintain and assure the integrity of the data base and to improve its performance via changes in its data structure.)

These activities were possible due to the unique design of IPSS which permits a top down modular view of information processing systems and the special language constructs and facilities developed specifically for the above purposes.

The IPSS system is designed to analyze information processing systems in their totality. Initially, IPSS mechanism for the definition and evaluation of hardware, computer system software and storage level data structures were developed and have been extensively tested. The next phase in the continued development of IPSS was to incorporate features for the description, manipulation and evaluation of logical data structures of the kind normally encountered in modern data base management systems. While the current IPSS capabilities substantially reduce the effort required to model system and data base level software, hardware and dataware, modelers can model all aspects of an information processing system. A high level query processing capability is currently being implemented and development is planned for a telecommunication specific extension.

3.0 BACKGROUND OF THE RESEARCH NEEDS

The main goals of the research are the extension of the current IPSS system to include language constructs and statistics attured to data base management system modeling activities, and to the measurement of system performance from a user perspective. The following discussion provides a background to evaluative methodologies associated with the research needs for these areas.

3.1 METHODOLOGIES FOR THE EVALUATION OF USER GOALS

As the complexity of current and anticipated systems increases, the analyst's need for more sophisticated evaluative techniques becomes more apparent. The types and number of services being provided by an information system are growing rapidly. Where once was a homogenous set of simiiar user classes which could be evaluated by relatively simple measures, now operates a multitude of user classes and services demanding various conflicting performance requirements and lacking a single common measure of performance. Thus, from an external performance point of view, the analyst must evaluate information systems against a multiple set of user criteria.

From an internal performance view, current and anticipated demands force the analyst to also evaluate multiple criteria. As the number of services provided increases, the size of the delivery system must also increase. This implies that the number of resources required to build the system will increase. Information system resources -- hardware, software and dataware are expensive to buy, install and operate. Thus, the analyst must also be concerned with the cost effective use of these resources.

Current evaluative technologies focus on only one criteria in the system design equation, either the use or the system's resource perfromance. The ability to simultaneously ascertain the impact of resource performance on user goals or vice versa is not readily achievable through these methodologies. One of the purposes of this evaluative

expansion is to establish a formal liaison between the evaluation of user goals as a function of system behavior and the analysis of resource performance as a function of user activity.

User oriented analyses with objective functions based on response time, throughput, and cost have been (are continuing to be) reported in the literature. Most frequently, analytic approaches use queueing models as their basis (GAV76, BUZ73, KLE76 are representative of this type of analysis). Due to the necessity to maintain tractable models, many simplifications are required for a model's analytical solution. Simulation models have also been applied to user oriented analysis (CON65, ROE70). Unfortunately, these models yield only average and/or aggregate measures of system response. As a result of these simplifications, the analyses produced by both of the approaches fails in many cases to identify the relationship between users and resources. Therefore, they are suspect when used to predict the impact on system performances of modifying the current environment.

Alternatively, performance analyses can be made from the system's standpoint, treating the user and his goals in the aggregate. The most common approach is a subsystem study, where a particular part of the information system complex is isolated, with the subsystem user(s) represented by a stochastic generator, both analytic and simulative. The most emphasized areas of research has been the I/O subsystem (ABA68, NAH73, BOU72) and CPU utilization (KLE72, HEL70, MLS76). The problem with this level of evaluation is that, although providing valuable local intuitive insight, these models rarely relate to the ultimate information system user, and, therefore, do not provide realistic insight into global performance.

Examinations of complete systems have also been made. Exhaustive hardware/software measurements have been analyzed by Lindsay and Cole (LIN76, COL72) while simulation models, including an aggregate user component, have been built by Reeves and Pooch, Norland, and Lum (REE75, NOR71, LUM70). Alghough results of the evaluations include resource utilization statistics and user oriented measures such as response time, there is little attempt in these models to relate particular resource

usage to the effort on user goal attainment. But from practical experience it is evident that there is indeed a relationship between user goals and resource usage. In fact, Buzen (BUZ76) has recently proposed some fundamantal laws for computer performance which relate resource activity to global system/user measures such as response time and throughput.

Thus, current evaluative efforts have not been able to provide an acceptable approach to handling the multiple criteria environment. Because of the inherent conflicting nature of the criteria an optimal design is not possible, and therefore, good system design is assumed to satisfice both performance related criteria. The approach embodied in this evaluative expansion was to first establish a casual relationship (liaison) between user goal attainment and system resource activity. Then after measuring the resultant behavior, a multi-objective technique was employed to produce a satisficed evaluation. The liaison will be specifically maintained in IPSS via the Task and Activity facilities. A mathematical programming technique, multiple goal programming (MGP) has been adapted to evaluate the multiple performance criteria (CHN77).

The importance of providing a task modeling capability is evident by examining current multi-programming system design philosophies. There has been a continuing trend toward hierarachic design of operating systems (DIJ68, ZUR68, MAD76, SDB77), both top-down and bottom-up. Such designs imply that particular functions are assigned to particular levels within the hierarchy, e.g., job scheduling at one level, hardware I/O control at another, etc. Each of the functions (levels) also in a sense, manage a class of resources via allocation, usage and de-allocation. In order to model and evaluate the effect of user demand on each of these functional levels, and thus, on their subordinate resources, one must be able to model the tasking function.

The overall modeling philosophy of IPSS, coupled with the addition of the TASK and ACTIVITY facilities allow the modeler to accomplish this. The purpose of the TASK facility is to maintain control over the resultant resource activity with respect to its causal user demand. This resource activity is further broken down by activities which characterize the individual system functions.

The control that the TASK facility exerts over its subordinate Activities and resources is essentially its maintenance of statistics. In IPSS the modeling resource usage automatically produces an extensive set of queueing and utilization statistics. The characterization of software processing also produces, automatically, service by service statistics on service time and hierarchy interactions. The TASK facility is associated with a particular user class to complete the user-resource liaison. It is the function of the TASK facility to maintain a set of statistics measuring the behavior of its subordinate resources. These statistics, collected dynamically, essentially partition of the aggregate queueing, utilization and service time statistics to each causal TASK (user class). Furthermore, the collection and maintenance of these statistics is performed automatically by IPSS. Thus, the first part of this approach to information systems establishes the link from user demand to resource activity and automatically produces statistics.

The second part of the approach to be implemented in this expansion is an evaluation of the multiple performance criteria. To be most effective, this evaluation should relate resource activity to user oriented performance goals. This entails the production of higher level performance measures, that may not be solely time related such as cost. This completes the two-way liaison between user demand and resource activity by measuring the contributions of resources to the satisfaction of user goals. This is, in part, accomplished via the TASK/ACTIVITY liaison, but the evaluation, however, is performed by a multiple goal programming based procedure.

The evaluative view of an information system formulated by the MGP procedure is identical to the modeling view characterized by the TASK and ACTIVITY facilities whereas using the TASK and ACTIVITY aids in the creation of a top-down modular performance evaluation (PE) model. In fact, the variables of the model are the TASK and ACTIVITIES. This analogy was designed to allow the automatic collection of statistics by the TASK facility during normal modeling be directly applicable to the PE model for evaluation.

The result of evaluating this PE model is two-fold. First, the desired evaluation of the user criteria is produced, on a global goal basis and on an individual goal basis (Task basis). Second, each activity is also evaluated relative to its service time performance, across all tasks (user classes). This evaluation can determine if a given activity is being under-utilized, saturated or used to its proper capacity, where proper is with respect to all tasks. These activity measures can be used as guidelines when deciding which designs/alternatives to follow. The most probable performance inhibiting activities can be identified in this way. An overview of the underlying methodology of this multiple criteria evaluation can be found in the paper by Chandler and DeLutis (CHN77).

3.2 METHODOLOGIES FOR THE EVALUATION OF DATA BASE MANAGEMENT SYSTEMS

Sable (SAB71) outlines a specific evaluation procedure for data base systems which is based on establishing the cost effectiveness of each candidate system with respect to a specific problem. He proposes a multistage method for "scoring" or grading DBMS features. The scores are normalized according to an assigned nominal requirement for each feature and multipled by a weighting factor which represents relative importance. The overall system score is the sum of the weighted results. This methodology relies heavily upon the intuition of the analyst to select the important factors and to assign correct weightings. In addition, it presupposes that the evaluation data are available.

While methodologies based upon this approach are useful for establishing guidelines and controlling the overall evaluation process, they are static in nature and therefore are not useful in measuring dynamic DBMS behavior. However, the current DBMS literature contains several methodologies specifically designed for the measurement of this type of behavior. These methodologies all provide a set of descriptive facilities for the definition of both DBMS software and computer hardware. In addition, they all provide a special purpose, discrete event simulation tool for the analyst/designer.

Phase II (OWE70) is one such tool for modeling hierarchical base management systems. Its objective is to allow the modeler to study the interaction of such DBMS variables as storage structure characteristics, data distribution, device interaction, and access methods. The modeler specifies the characteristics of the data, the data accessing strategies, and the underlying hardware configuration through a special purpose user-oriented language. Data field contents are characterized by distribution functions and data fields are lelated to each other to form what is called segment (the logical structure of the data). The logical data structure is then mapped to a physical storage structure by partitioning the records into files which exist on hardware devices. The accessing strategy allows the modeler to specify the lists of records which are to be accessed, the accessing method (i.e., sequential, indexed, or direct), and the order in which the accesses are to occur. Phase II is limited to the I/O analysis of single level hierarchical data base management systems. Its design objectives are intentionally modest and thus it is not suitable for modelling more complex software interactions.

A methodology for evaluating data base scheme, physical data organization, and data base loading is described by Hulten and Soderlund (HUL76). Their simulator is called ARTS/DB (Analysis of Real Time Systems/Data Base oriented systems). It is implemented in Simula and executes on the DEC10 system. The descriptive facilities which they provide allow the characterization of data base systems in terms of CODASYL-DBTG features. An ARTS/DB model consists of four levels, i.e., definitions of the application, the DBMS, the system software, and the hardware. This provides the analyst with a useful mechanism for performing successive experiments with only slight modifications to the model.

A data base simulator based on the Data-Independent Accessing Model I (DIAM I) (SEN72) has been developed with two ojectives in mind: the comparison and selection of existing data base systems and the design and evaluation of new data base management techniques

(SCH76). This simulator consists of four subsystems similar to the ARTS/DB approach. In this case the subsystems correspond to the levels of the DIAM and are: a query generator, a query translator which uses defined access paths, and I/O generator, and the host computer system.

A three stage DBMS model building approach as well as a simulator system utilizing this approach has been described by Reiter and Finkel (REI75, REI76). First, high level user-supplied logical data view descriptions are translated into an intermediate standard form. Second, an instruction stream is generated which refers to data by block address. In this stage no assumptions are made concerning the location of data in physical storage. Third, the dynamic elements of the system are employed to produce performance measures. The particular DBMS under study is parameterized with respect to data representation, resulting in a specific model of that DBMS. The model is constructed of static and dynamic parts which results in a similarity to operational DBMS and also in executional efficiencies.

DeLutis (DEL72) presents a methodology for the analysis of auxiliary storage systems (AUXSIM). In this system the modeler describes hardware devices and software processes using language statements which are highly descriptive of these devices and processes. The AUXSIM translator produces Fortran source code which executes as a discrete event simulation of the system. This research was the predecessor to the Information Processing System Simulator (IPSS). The research proposed here was, in part, a further extension of the IPSS methodology to data base management systems.

In addition to these generalized evaluation techniques, several specific DBMS models have been developed using a variety of simulation languages and techniques. The primary evaluation criteria used by these simulations is response time and throughput.

Ghosh and Tuel (GHO76) designed an experiment to model a large IMS data base to determine the effect of varying such DBMS features as the retrieval sequence and logical data structures. They collected operational data by executing IMS in a stand-alone environment. Their measurement criteria was the elapsed time required to execute a

sequence of calls. Griffith (GRI75) constructed a GPSS simulation model of Univac's DMS-1100 also using response time as the evaluation criteria. The simulation results reported by Nakamura (NAK75) include response time, resource utilization (cpu, channel, disk), as well as statistics related to transactions and queues. Hall (HAL74) describes a DBMS simulation model of a hierarchical data base whose objective was to minimize response time and costs to the user.

4.0 EXPERIMENTS

A series of modeling experiments were performed between mid June and mid September 1977 for the purpose of; (a) assessing the ease of employing the IPSS methodology and language in modeling complex systems, and (b) evaluating the usefulness of the proposed IPSS language and statistics extensions. The system modeled was an on-line real-time system comprised of; (a) data entry terminals; (b) a host main frame for the application software and user interface; and (c) a CODASYL-like data base residing on a backend mini computer. Section 4.1 summarizes the modeling activities. (The reader may refer to Appendix A for a complete discussion of each of these models.) The following subsection provides background as to why such a complex system was chosen.

The purpose of the experiment, initiated by the U.S. Army Computer Systems Command, was to evaluate a complex information system architecture via the Information Processing System Simulator (IPSS). The primary objectives of the experiments were three-fold:

1. To demonstrate the capability of IPSS and its underlying methodology to accurately model a complex system,
2. To ascertain its ease of use, and
3. To develop a generalized basic model of the following systems:
   a. an interactive version of the SIDPERS Personnel system resident in an idealized Host computer, and
   b. The IDMS data management system resident in a PDP-11/70 serving as the back-end computer;

   and through the model judge the appropriateness of the proposed IPSS language and statistics extensions.

Results indicate that IPSS is well suited for the task. Through the use of a top down modeling philosophy a complete model of the application and IDMS processing was written and tested in less than two man months and model statistics tracked well to the available performance measures. The following subsection provides an overview of the salient features of the modeling effort.

4.1 THE MODELING EFFORT

The purpose of the modeling activities was to demonstrate the feasibility and suitability of the IPSS to model large, complex information processing systems. Reflection on the modeling activities associated with the SIDPERS/IDMS experimentation provides positive evidence that the IPSS is equal to the task. One important reason that IPSS achieves its goal is that its underlying methodology provides a top down, modular approach to system analysis and model synthesis thus facilitating answers to complex analysis and design questions.

Several models were constructed, each of which reflected system details suited to the multi-stage development process which was adopted. The ease of model expandability in both breadth and depth was demonstrated through the construction of independent models and their integration into an overall complete model. Appendix A summarized their major features. The SIDPERS/IDMS workload model represents the software processes of the applications and the IDMS in detail while characterizing the operating systems and computer hardware as a block box. After this basic model was tested, subsequent models were developed by splitting the software processes at the host/back-end interface and adding characterizations of the appropriate hardware and operating system software functions. The synthesis of the final overall model was easily accomplished since the system interfaces were clearly identified.

These models were independently tested, each producing a set of statistics reflective of the modeled processes and computing environment. These results are summarized in Section B.2. In all cases queueing and utilization statistics were obtained for the modeled software and hardware resources. These statistics provide an easily understood snyopsis of activities since one or more IPSS services were used to represent major units of processing (e.g., application programs, TP line scheduling, IDMS processing). Statistics were also automatically collected on each of the hardware facilities indicated.

The modeling activities were performed with approximately six man months of effort. All the models to be described in the next section were completed. However, three conditions occurred which prohibited the creation of the sought-after final models. These conditions were:

1. The IPSS/DBS parsers were not completed in time for use. However, models of the data base reflecting the IDMS scheme have been written in the IPSS/DBS syntax and semantics specifications. Furthermore, these submodels have now been executed under the IPSS/DBS.
2. Combination of all the submodels (excluding the IDMS schema) could not be successfully compiled. Investigations into the cause of the problem has lead to the detection of an error in the IPSS nucleus. The nucleus did not detect an IPSS library manager message signaling the lack of library space. Thus model code was lost and was undetected until final model assembly. This condition has been recently fixed and the models have been successfully executed.
3. The interfacing mechanism discussed in Appendix C3 has not been completely implemented. This is due to the unexpected quantity of code which had to be modified. Therefore, the IPSS and IPSS/DBS cannot currently execute together. This problem is currently being aleviated.

Section 4.2 discusses the complete modeling program envisioned for the modeling experiment. Appendix B discusses the statistics for the submodels actually completed during the experiment.

4.2 SIDPERS/IDMS MODELS

The purpose of this section is to provide an overview of the system analysis and model design activities related to the IPSS modeling of the interactive SIDPERS/IDMS system.[7] For analysis purposes the system was divided in the following logical model components:

---

[7] The IPSS methodology provides a modular, top-down approach to system analysis and model synthesis thus permitting the solution discussed in this section.

Hardware

Host machine configuration

Back-end machine configuration

Software

Operating system for the host machine

Operating system for the back-end machine

Terminal interaction modules

SIDPERS transactions

Interface modules

IDMS

These logical model components were used to define five inter-connected system activities each representing specific SIDPERS/IDMS functions. This organization is shown in Figure 4.2-1. A number of models were written which incorporated a one or more of the modeling components and represented one or more system activities. Figure 4.2-2 identifies the submodels and incorporated model components.

Because of the magnitude and complexity of the system, a phased approach was taken in model development. The model synthesis and validation is implicit in IPSS, thus, the approach posed no difficulties. In Phase I a model of the SIDPERS transactions as processed by IDMS (system activities 1 and 5) was constructed. However, it is important to note that the entire structure of the final model was represented in this preliminary model. This was accomplished through IPSS services for system activities 2, 3, and 4 which were essentially null routines. That is, they were invoked by other services but their internal processing function remained unspecified. In particular, terminal interaction and secondary storage accesses were characterized merely by advancing the simulator clock. This simplified approach allowed the SIDPERS transactions and the IDMS processing functions to be modeled and validated quikly. Once this workload flow model was executing successfully and generating statistics, the operating system and computer hardware model components were added in succeeding modeling phases.

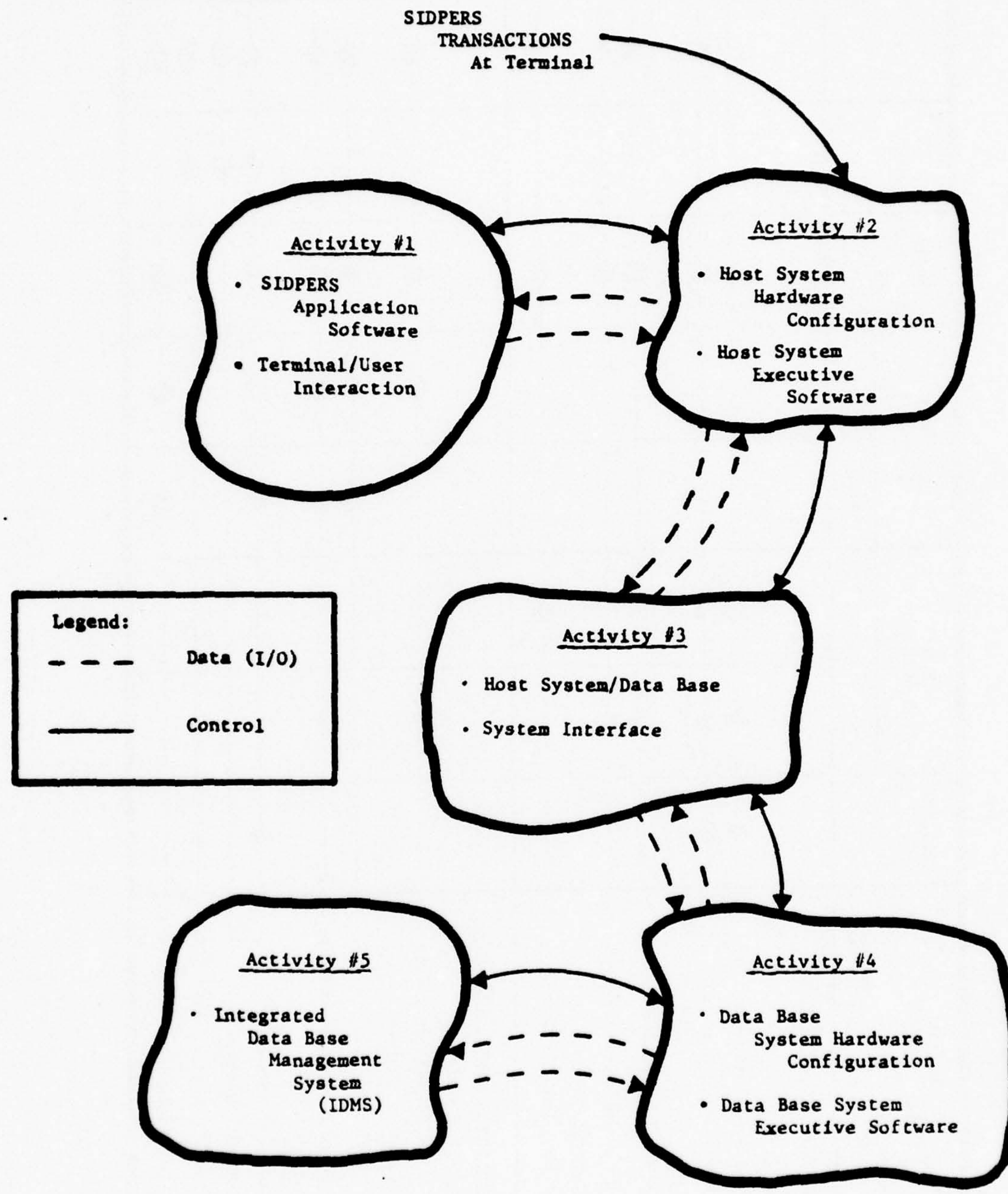


Figure 4.2-1 Structure of Final SIDPERS/IPSS Model

Models / System Facilities Represented	SIDPERS/ IDMS Workload Model	Host Submodel V1	Host Submodel V2	Back-end Submodels V1	Back-end Submodels V2	Host/ Back-end Models	IDMS	Total
<u>Application Environment</u>								
<u>Activity #1</u>								
● IDPERS Application Software	●	●	●			●		
● Terminal User Interface	●	●	●			●		●
<u>Activity #2</u>								
● Cost System Hardware		●	●			●		●
● Cost System Executive Software		●	●			●		●
<u>Activity #3</u>								
● Cost to PDP 11/70 Interface			●			●		●
<u>Data base Environment (PDP 11/70)</u>								
<u>Activity #3</u>								
● PDP 11/70 to Host Interface					●	●		●
<u>Activity #4</u>								
● PDP 11/70 Hardware				●	●	●		●
● PDP 11/70 Executive Software				●	●	●		●
<u>Activity #5</u>								
● DMS General Software	●			●	●	●	●	●
● DMS DML Statements							●	●
● Data Base - Logical Structure							●	●
● Data base - Physical Structure				●	●	●		●

Figure 4.2-2 IPSS SIDPERS/IPMS Models

Phase II focused on the host environment. The base model was system Activity #1 as represented in the Phase I model with the backend treated as a black box. To this base model was added successively more detailed characterizations of the host system. The final model in this phase included the characterization for the backend interface hardware and software. A synopsis of Phase II activities is given below. Appendix A2 presents detail consideration of the modeling activities in this phase.

Phase III was concerned with the characterization of the backend system. Again the modeling proceded in an evolutionary manner. A sequence of models was planned which would add the PDP 11/70 hardware and operating system software to the IDMS workload model developed in Phase I. For these activities the front end was treated as a white box, i.e., a external source of back end requests. Specifically omitted was any attempt to model IDMS DML's or traversal activities for the SIDPERS logical data base (as defined via the IDMS DDL's). These latter functions were modeled using the IPSS/DBS facilities. A synposis of this phase is given below. Details are found in Appendix A3.

The goal of Phase IV was to combine the Phase II and III models into a single model. As mentioned at the begining of this appendix, a minor IPSS programming error, since corrected prevented their phase to be completed within the scheduled experimentation time line. Results of this modeling phase will be reported at a latter date.

Phases V was planned whose purpose was to model the IDMS DML's and the DDL described SIDPERS data base. A paper model was completed. However, execution was delayed until completion of the IPSS/DBS parsers. As with Phase IV the activity was completed after the termination of the modeling activities. This model is being reported in Mr. Brownsmith's disseration, titled "A Methodology for the Performance Assessment of Data Base Systems: An Extension to the Information Processing System Simulator Methodology", which is to be available Spring, 1979 from Ohio State University.

The goal of Phase VI was to integrate the Phase V model with the final Phase IV model. Again this activity has to await modifications to the current IPSS and IPSS/DBS.

A Synopis of the Phase II Activities (SIDPERS/HOST Models)

The SIDPERS application model was built assuming a computer free environment. The purpose of the Host submodel was to provide both the hardware configuration and the operating system of a host computer. Since no particular computer system had been identified as the host computer for the SIDPERS/IDMS project, a generalized host model was constructed. The elements of this model were characterizations of those operating system functions that support the SIDPERS application and the Host/Back-end interface. These include the job scheduling, task management and resource allocation, assumed to be available in the Phase I model, were now characterized. The hardware configuration that was modeled consisted of only these hardware components required to support the modeled host software.

The approach to construction of the host model followed the premise that the host operated in a reactive mode. The existing SIDPERS model was analyzed and when an operating system function was required, a module that characterized its behavior was added to the Phase II models. This procedure was followed for all levels of applications represented in the SIDPERS 1, and for the Host/Back-end interface needs. As a result a model was created of a host computer environment that satisfied the SIDPERS operating needs. The operating system functions included in the host model were characterized in a general, modular parameterized manner, such that at a later time, the mechanisms of the functions of a particular host computer could be easily incorporated into the existing model. Thus, although system activities 1 and 2 were communicating continuously, modifications to the latter could be made without affecting the existing SIDPERS application model. The SIDPERS/HOST submodels were tested by "black boxing" the back-end machine.

Synopsis of Phase III Activities (BACK-END/IDMS Models)

The IDMS model was built like the SIDPERS application model, assuming no particular computer environment. The purpose of the Back-end system

model was to provide the hardware configuration and the operating system of the back-end environment. Previous Army studies had designated the PDP-11/70 as the specific computer to be employed as the back-end computer. The functions of this computer system that needed to be modeled were the operating system functions to support IDMS and Host/Back-end interface routines. As with the host submodel, the back-end submodel was assumed to operate in a reactive mode, responding to requests for its services. The hardware configuration that was modeled was the complete USACSC PDR-11/70 configuration, recently moved with AIRMICS to the Georgia Institute of Technology in Atlanta.

The back-end models were built modularly, adding chracteristics of operating system or interfacing functions as they were needed. Although these characterizations were made specific to the AIRMICS PDP-11/70 computing environment, they were modeled in a parmeterized, modular manner so that if different PDP-11/70 strategies were to be substituted, the IDMS and interface submodels would not be affected. The IDMS/Back-end models were tested using the SIDPERS application model as the work load generator.

4.3 INSIGHTS GAINED FROM THE MODELING

The purpose of the modeling effort reported in this document was to demonstrate the feasibility and suitability of the Information Processing System Simulator (IPSS) with respect to modeling large complex information systems. The system modeled to demonstrate these capabilities was a host/back-end processor configuration which supported interactive application processing and a data base management system. The modeled software processes included a detailed characterization of application loading, DBMS processing, and the operating system functions of task management, job scheduling and resource allocation.

The IPSS methodology provided a modular, top-down approach to system analysis and model synthesis thus facilitating a solution to this complex problem. Several models were constructed, each of which

reflected system details suited to the multi-stage development process which was adopted. The ease of model expandability in both breadth and depth was demonstrated through the construction of independent models and their integration into an overall complete model. The SIDPERS/IDMS workload model represents the software processes of the applications and IDMS in detail while characterizing the operating systems and computer hardware as a black box. From this basis model, two subsequent models were developed by splitting the software processes at the host/back-end interface and adding characterizations of the appropriate hardware and operating system software functions. The synthesis of the final overall model was easily accomplished since the system interfaces were clearly identified.

These models were independently tested, each producing a set of statistics reflective of the modeled processes and computing environment. These results are summarized in Appendix B. In all cases queueing and utilization statistics were obtained for the modeled software and hardware resources. These statistics provide an easily understood synopsis of activities since one or more IPSS services were used to represent major units of processing (e.g., application programs, TP line scheduling, IDMS processing). Statistics were also automatically collected on each of the hardware facilities indicating both potential bottlenecks and under-utilized components.

These models were all internally verified. That is, through experimentation, the internal processing consistency was verified to be accurately reflective of the real world processes. Due to the lack of operational data, however, it was not possible to validate them through comparison to actual system performance. Thus, although some confidence can be placed in the relative values of the statistics reported in the Appendix B, no validation has been made with respect to the magnitude of the data values.

This project has demonstrated the suitability of IPSS to model complex systems while requiring a minimal time for model synthesis. The modeling effort as reported here required approximately four man months. The resulting models have proven to be versatile as well as

adaptable to changing requirements indicating that significant portions can be used for other application areas. For example, the DBMS and operating system processes as well as hardware facilities need not be modified when another major application area is modeled, thus resulting in a substantial reduction in future related modeling efforts.

In addition, this project has served to focus future development work as well as to validate current goals. In particular, this project has demonstrated the need for simulation facilities specifically attuned to the characterization of integrated data structures as represented in IPSS/DBS. These models did not use these features since they were not fully operational, and thus the logical structure of the data base was charcterized at a more abstract level then desired. Furthermore, experimentation with record mapping policies was not attempted because of the complexities involved. Finally, several other research endeavors were identified. In particular, large complex models of computer networks could easily be constructed from models of individual nodes if a multi-simulator approach were adopted. Work is currently underway to allow multiple node simulations through extension of IPSS facilities and concepts. Extensions for modeling teleprocessing activities are also planned.

5.0 FUTURE EXTENSIONS TO THE IPSS

This research activity is just another step in a continuing IPSS research, development and engineering program whose goal is to create a comprehensive computer aided design facility appropriate to all stages in an information system's lifecycle. Three steps in the RD&E plan have now been completed. These include

1. 1967 - 1972: Initial Methodological and Simulator Development - this activity resulted in the Auxiliary Storage System Simulator (AUXSYM). Details are described in "A Methodology for the Analysis of Auxiliary Storage Systems" available from University Microfilms.
2. 1972 - 1975: Extension to the methodology and complete redesign of the simulator. These activities resulted in the IPSS; details are reported in "A Methodology for the Performance Evaluation of Information Processing Systems" (DEL77) and "The Information Processing System Simulator (IPSS): Language Syntax and Semantics".
3. 1975 - 1977: Continued methodological development to include the following
   a. DBS specific features;
   b. Assessment of system effectiveness with respect to competing user goals;
   c. Addition of mechanisms to determine costing policies.
   d. Assessment of system loading created by multi key query processing
   e. Development of comparative evaluative techniques for simulation languages vis-a-vis performance assessment needs.
4. 1976 - 1979: Verification and Validation - The purpose of these activities has been to demonstrate the effectiveness of the IPSS as a modeling facility. Validation has included paper models of different vendor hardware

software and file structures, models of idealized systems, models of existing complex systems (this report describes one such system.) A proposal by Chandler is designed to extend this activity into more specific validation activities where extrapolation can be validated.

As a result of the current and past research experiences, an extensive number of continuing research and experimental activities have been identified. They are presented here to illustrate the potential for the IPSS and its underlying methodology and to identify a means of activities which must be accomplished in order to achieve the ultimate research objective - that being to create a computer assisted design facility.

A. Extensions to the IPSS Methodology

The goal of these research activities is to continue the extension of the methodology's scope of applicability to (a) higher views of information system processes (b) other stages in the information process system lifecycle and (c) analysis and design processes. Five research activities have been identified for this area. They are:

1. Telecommunications - extension of the IPSS the permit characterization of data communication subsystems in a manner familiar to their structure and behavior.
2. Requirement Analysis - extension of the IPSS to interface with requirements analysis methodologies such as Tiechrow's PSL/PSA and TRW Corp's SCREM.[8]
3. Sensitivity Analysis - development of a methodology which permits the assessment of models to identify performance sensitive components.
4. Estimating System Lifecycle Costs - development of a methodology to examine IPSS models for the purpose of (a) estimating the number and complexity of components comprising the modeled system, and (b) using these data as input to lifecycle estimating techniques.

[8] PSL/PSA is acrynon for Problem Statement Language/Problem Statement Analyzer SREM stands for System Requirement Engineering Methodology.

5. Automated Component Selection - formulates a methodology which is based on a predefined view of system components, and on one or more design criteria will automatically alter IPSS models thus identifying feasible designs for additional analyses by the modeler.

B. Extensions to the IPSS Language

The goal of this development activity is to expand the model synthesis and statistical display facilities of the current IPSS in order to reduce modeler effort. Five development activities have been identified. They are:

1. Statistics Generation - development of language extensions to cause the automatic gathering of standard statistics for sequential ("serially reusable") processes.
2. Verification and Validation - extends existing and develop new statements to support model verification and validation activities.
3. Operating System Characterizations - extend existing and develop new facilities attuned to; (a) the modeling of information system job scheduling, resource allocation and task management functions and (b) characterizing main memory management, CPU interrupt activity and CPU instruction times.

C. Extensions to IPSS Simulation Facility

The purpose of these activities is to increase the ease of use of the IPSS simulator facility by the modeler. The following four items are proposed as extensions in this area:

1. On-Line Modeling - extend the processing environment for the IPSS to include; (a) on-line entry of IPSS language statements comprising an IPSS input stream, and (b) an on-line, real-time prompter facility to guide the modeler in the use of IPSS language syntax. The modeler will be able to switch between model stream definitions and stream processing as desired.

2. MACRO Library - the IPSS macro and library facilities currently permits the modeler to retain models and/or model components for future reference. This activity would develop a set of generalized structures presenting the salient features of the following software:
   a. Operating System: task management, job scheduling, resource allocation and memory management;
   b. File Utilities: sort/merge, file copy, and opening and closing of files;
   c. Data Base Management System: buffer management, high level DML's, schema structures.
3. Graphical Display - the IPSS would be enhanced to permit graphical interaction between the system and the modeler. Initially this will involve only output of items such as:
   a. Model statistics including frequency listographics.
   b. Model structure showing the network of services described for the model and the data and file structures.
   c. Dynamic model structure showing the sequencing of service invocation and their hierachy during a task's execution.

Ultimately, the goal of graphics use is to prove a modeler/system communication interface which will permit interactive support of model synthesis and the display of model structure, model verification data, and model behavior statistics.

REFERENCES

ABA68 Abate, J., Dubner, H., and Weinberg, S.B., "Queueing Analysis of the IBM 2314 Disk Storage Facility", JACM, Vol. 15, No. 4, (1968), pp. 577-589.

BOU72 Boutross, King and Rutledge, "Discrete Simulation Model of DASD", IBM Research (RC3698), Jan, 1972.

BUZ73 Buzen, J.P., "Computer Algorithms for Closed Queueing Networks with Exponential Servers", CACM, Vol. 16, No. 9, (1973), pp. 527-531.

BUZ76 Buzen, J.P., "Fundamental Laws of Computer Performance", in Proceedings of International Symposium on Computer Performance Modeling, Measurement and Evaluation (CPMME), (1976) pp. 200-210.

CHN77 Chandler, J.S. and DeLutis, T.G., "A Multi-Criteria Approach to Information Systems Design", to be published in Proceedings of National Computer Conference, (1977).

COL72 Cole, Gerald, D., "Performance Measurements on the ARPA Computer Network", IEEE Transaction on Communication, (1972) pp. 630-636.

CON65 Conger, C.R., "The Simulation and Evaluation of Information Retrieval Systems", Report 352-R-17, (April 1965).

DEL72 DeLutis, T.G., "A Methodology for the Analysis of Auxiliary Storage Systems", Ph.D. Dissertation, Purdue University, 1972.

DIJ68 Dijkstra, E.W., "The Structure of 'T.H.E' Multiprogramming System", CACM, Vol. 11, No. 5, (1968), pp. 341-346.

GAV76 Gaver, D.P., and Hunfeld, G., "Multitype Multiprogramming: Probability Models and Numerical Procedures", Proceedings of GPMME, (1976), pp. 38-43.

GHO76 Ghosh, S.P., and Tuel, W.G., "A Design of an Experiment to Model Data Base System Performance", IEEE Transactions on Software Engineering, Vol. SE-2, No. 2, (1976).

GRI75 Griffith, W.G., Ingerman, D., and Price, C.E., "A Simulation Model of Univac's D,S-1100 -- More than Just a Performance Evaluation Tool", Symposium on the Simulating Computer Systems, III, (1975), pp. 91-99.

HAL74 Hall, W.A., "A Simulation Model to Aid in the Design and Timing of Hierarchical Databases", Winter Simulation Conference, (1974), pp. 277-284.

HEL70 Hellerman, H.R., and Smith, H.J., "Throughput Analysis of Some Idealized Input, Output and Computer Overlap Configurations", Computing Surveys, Vol. 2, No. 2, (1970), pp. 111-118.

HUL75 Hultén, C., and Söderlund, L., "A Tool for Performance Evaluation of Data Base Oriented Information Systems", Project MAD, Research Group CADIS, Department of Information Processing and Computer Science, The Royal Institute of Technology and the University to Stockholm, Sweden, (1975).

KLE72 Kleinrock, L., and Muntz, R.R., "Processor-Sharing Queueing Models of Mixed Scheduling Disciplines for Time-Shared Systems", JACM, Vol. 19, No. 3, (1972), pp. 464-482.

KLE76 Kleinrock, L., Queueing Systems, Volume 1: Theory and Volume 2: Computer Applications, New York: John Wiley & Sons, Inc, (1976).

LIN76 Lindsay, D.S., "A Hardware Monitor STudy of a CDC KRONOS System", Proceedings of CPMME, (1976), pp. 179-186.

LUM70 Lum, V.Y., Ling, H., and Senko, M.E., "Analysis of a Complex Data Management Access Method by Simulation Modeling", FJCC, (1970), pp. 211-222.

MAD74 Madnick, S., and Donovan, J., Operating Systems, New York: McGraw-Hill, 1974.

MLS76 Mills, P.M., "The Overlapping of CPU and I/O Processing in Multiprogramming", in Proceedings of CMG-VII, (1976), pp. 200-206.

NAH73 Nahourii, E., "Direct Access Device Simulation", IBM Systems Journal, Vol. 13, No. 1, (1973), pp. 19-31.

NAK75 Nakamura, F., Yoshida, I., and Kondo, H., "A Simulation Model for Data Base System Performance Evaluation", National Computer Conference, (1975), pp. 459-465.

NOR71 Norland, K.E., and Bulgren, W.C., "A Simulation Model of GECOS III", Proceedings of ACM, (1971), pp. 596-612.

OWE70 Owens, P.N., "Phase II: A Data Base Management Modeling System", Proceedings of IFIP, (1970).

REE75 Reeves, T.E., and Pooch, U.W., "A Multiple Subsystem Simulation of Processor Scheduling", Symposium on the Simulation of Computer Systems, III, (1975), pp. 129-135.

REI75 Reiter, A., "On Performance Modeling of Data Base Management Systems -- An Inductive Approach", MRC Technical Summary Report, Mathematics Research Center, University of Wisconsin, Madison, (1975).

REI76 Reiter, A., and Finkel, B., "Simulating a Virtual Data Machine", MRC Technical Summary Report, Mathematics Research Center, University of Wisconsin, Madison (1976).

ROE70 Roehrkasse, R.C., and Smith, D., "Simulation of Operating Systems", Technical Report GITIS-70-11, (1970), Georgia Institute of Technology.

SAB71 Sable, J.D., "Generalized Data Management Systems: Query and Language Processing", Proceedings of the International Seminar on Information Storage and Retrieval, (1971), pp. 93-111.

SCH76 Schneider, L.S., and Connolly, T.W., "A Generalized Database System Simulator Based on the Data Independent Accessing Model I", Winter Simulation Conference, (1976), pp. 513-524.

SDB77 SIGBDP, "Guidelines for Cost Accounting Practices for Data Processing", DATA BASE, Vol. 8, No. 3, 1977.

SEN70 Senko, M.E., Altman, E.B., Astrahan, M.M., Fehder, P.L., and Wang, C.P., "A Data-Independent Architecture Model 1: Four Levels of Description from Logical Structures to Physical Search Structures", IBM Research Report RJ982, San Jose, California, February 1972.

ZUR69 Zurcher, F.W., and Randall, B., "Iterative Multi-Level Modeling: A Methodology for Computer System Design", IFIP 68, pp. 867-871.

ADDITIONAL REFERENCES

DEL76 Delutis, T.G., and Smith, J.D., "IPSS/DBMS: A Simulator for Evaluating DBMS Performance", Winter Simulation Conference, 1976, pp. 527-536.

DEL77 Delutis, T.G., "A Methodology for the Performance Evaluation of Information Processing Systems", Final Report to the National Science Foundation, Grant Number GN 36622, 1977, Available via NTIS.

REFERENCES USED IN CONJUNCTION WITH MODELING EXPERIMENTS

Cullinane Corporation, IDMS Data Base Design and Definition Guide, Release 4.0, June 1976.

Cullinane Corporation, IDMS DDL Data Definition Languages, Utilities and GCI Reference Guide, Release 3.1, Revision 1, April 1975.

Cullinane Corporation, IDMS DML Data Manipulation Language Programmer's Reference Guide, Release 3.1, April 1975.

Cullinane, J., Goldman, R., Mueurer, T., Nawara, R., "Commercial Data Management Processor Study", Cullinane Corporation, Wellesley Mass., December 1975.

Digital Equipment Corporation, IAS User's Guide, DEC-11-OIUGA-B-D, 1976.

Digital Equipment Corporation, Microprocessor Handbook, 1976.

Digital Equipment Corporation, PDP11 IAS/RSX-11 I/O Operations Reference Manual, DEC11-OIORA-A-D, 1975.

Digital Equipment Corporation, PDP11 Peripherals Handbook, 1976.

Digital Equipment Corporation, PDP11/70 Processor Handbook, 1976.

Digital Equipment Corporation, PDP-11 Computer Family Products and Services, 1976.

Hunter, F., Personal Communication, June 1977.

Hunter, F., Mercier, R., Peckham, G., Sernovitz, L., "Report on the Conversion of a File Oriented Interactive System to an Interactive Data Base Management System", U.S. Army Computer Systems Command, Fort Belvoir, VA, 1977.

Mercier, R., Mills, R., Price, J., Sernovitz, L., "Report of an Experimental On-line System", U.S. Army Computer Systems Command, Fort Belvoir, VA 1976.

Schaff, H., "Description of the VIABLE Transactions for the DIMUI Model", U.S. Army Computer Systems Command, Fort Belvoir, VA, 1976.

PRECEDING PAGE BLANK-NOT FILMED

APPENDICES

APPENDIX A

BACKGROUND TO THE MODELING EXPERIMENTS

The objective of this experiment was to provide a realistic problem environment against which the IPSS design facility (simulator and methodology) could be applied. The system modeled was chosen by Mr. Allan Curry of AIRMICS* specifically to test IPSS's design capabilities, ease of implementation, and adequacy and accuracy of the resultant statistics. The chosen system architecturally is of great interest to the U.S.A. Computer Systems Command as a replacement for exists BASOP systems. Section A1 of this appendix discusses the problem environment including a full breakdown of the hardware and software characteristics of the system. Section A2 defines the IPSS models generated to characterize the system.

A1. THE PROBLEM ENVIRONMENT

The purpose of the following discussion is to provide a background to the system modeled. The system modeled evolved from a series of R&D efforts performed by the U.S. Army Computer Systems Command (USACSC) at the request of the Department of the Army, Director of Management Information Systems (DA DMIS) in direct support of their "Vertical Installation Automation Baseline" (VIABLE) Project. The VIABLE project was established to upgrade both hardware and software at Army data processing installations. Included in its scope is the competitive procurement of new ADP equipment, and the extension of functional software capabilities to include management information system requirements as well as interactive processing.

The initial USACSC research plan was developed in September 1975. Its purpose was to explore and demonstrate those technologies which would conceivably be utilized in Army data processing systems developed

*AIRMICS stands for Army Institute for Research in Management Information and Computer Sciences.

over the next five to eight years. These areas included on-line processing, formal data base management systems, mini and micro computers and distributed data processing systems.

One aspect of the plan was to select a single functional area and build a baseline system which could be modified repetitively to add new features and migrate from one environment to another. The Standard Installation/Division Personnel System (SIDPERS), which processes Military Personnel actions was selected.

The computer system modeled was composed of a Front End Module (host machine) which processes SIDPERS transactions in a real-time interactive mode, attached to a Back End Module utilizing Cullinane Corporation's Integrated Data Base Management System (IDMS) to interface with the personnel data base. The backend machine modeled was the DEC PDP-11/70 System.

A1.1 SIDPERS

SIDPERS (Standard Installation/Division Personnel System) is the batch oriented multi-command personnel data processing system. It is a large system, consisting of approximately 800 types of transactions which perform update and retrieval operations on a personnel data base. However, since six of these transactions account for almost eighty percent of the total transaction activity, only these transactions were selected for incorporation into the baseline system. These transactions perform the following functions: adding a soldier to the data base, establishing duty status when a soldier arrives at an installation, changing the duty status and grade of a soldier, deleting a soldier from the data base on departure from an installation and acquiring data on a soldier and/or unit. These commands are denoted ADD, ARRIVAL, DUTY CHANGE, GRADE CHANGE, DEPARTURE, and INQUIRY respectively.

The SIDPERS test system required the creation of a small data base of sample records. Approximately 4,500 records representing 12 different

record types were selected and organized into an IDMS processable data base. The details of this data base and its organization are given in the next section.

The SIDPERS test system facilities for interactive transaction processing have been described by Schaaf (SCH77)[1]. Basically, special terminal interface processing modules were designed which allowed a terminal user to enter the above transactions in either a batch or tutorial mode of operation. In batch mode, all the data for a complete transaction is entered at once. In the tutorial mode, the terminal user is prompted for every data item of the transaction. This requires an access to a VALIDITY record which contains the text for prompting. Items that fail to pass the validity check are returned to the terminal operator for re-submittal. For either mode of operation, once all the data items of a transaction pass this test, the corresponding application program is invoked. Essentially, there is one application program for each of the transactions listed above. As part of application processing, compatibility checks are performed. If a data item doesn't pass this check, several accesses are required to correct the error; namely an access to a COMPATIBLE record, and several accesses to VALIDITY records. Such errors cannot always be resolved and in these cases the transaction is terminated. Otherwise the application program issues a series of DML commands to the IDMS Modules in order to store, modify, and retrieve records in the data base. During the IDMS Processing of each DML, application processing is suspended. When the application is completed, control is returned for further interaction.

A1.2 Prior Analysis Activities of SIDPERS

The data required to characterize these activities was provided from a previous study of VIABLE transactions (SHA77). (See Tables A1.2-1 and A1.2-2 for further specifics). This document provided the following data for each of the six types of transactions:

---

[1] NOTE. All references are found at the end of the main body of this report.

Table A1.2-1 The Results of the SIDPERS Interaction Test

	No. of entered data items	Data no. of transactions	No. of Cancelled transactions	No. of Transactions in Direct mode	No. of transactions in tutorial mode	No. of validity errors	% of data items with val. errors	No. of compatibility errors	% of data items with comp. errors
ARRIVAL	9	158	25 / 16%	93 / 59%	65 / 41%	29		31	
DEPARTURE	12	193	34 / 18%	111 / 57%	82 / 43%	60		101	
DUTY CHANGE	8	331	23 / 7%	208 / 63%	123 / 37%	91		170	
GRADE CHANGE	10	52	11 / 21%	31 / 60%	21 / 40%	15		34	
ADD	70	6	0 / 0%	0 / 0%	6 / 100%	17		6	
INQUIRY	3	38	0 / 0%	31 / 82%	7 / 18%	2		12	
TOTALS		778	93 / 12%	474 / 61%	304 / 39%	214	2.9%	354	4.8%

Table A1.2-2 Average Entering Time

Transaction Type	Entering Time Per Transaction		Transaction Characteristics		Entering Time Per Data Item	
	Tutorial Min:Sec	Direct Min:Sec	No. of Entered Data Items	Errors (about 8%)	Tutorial (sec)	Direct (sec)
ARRIVAL	5:23	2:49	9	1	32.3	16.9
DEPARTURE	6:03	2:55	12	1	27.9	13.7
DUTY CHANGE	3:58	1:40	8	1	26.4	11.1
GRADE CHANGE	5:55	1:48	10	1	32.3	3.8
ADD	25:57		70	6	20.5	
INQUIRY	1:23	0:54	3	-	27.7	18
AVERAGE OVER ALL TRANSACTIONS					27.9	13.9

1. the relative frequency of occurrence,
2. the percent direct mode,
3. the number of data items per transaction entry,
4. the percent data item validity error,
5. the percent data item compatibility error,
6. the time for direct mode data items,
7. the time for turorial mode data items, and
8. the percent of transactions canceled.

This data was incorporated into the IPSS model of SIDPERS/IDMS through Fortran DATA statements and was used to "drive" the model through representative terminal sessions. In addition, this document contains for each transaction, a list of the DMLs that are invoked and their sequence within the application program. This data was used to invoke lower level DBMS processes as described in the next section.

A1.3 IPSS Model of SIDPERS

The IPSS SIDPERS/IDMS transaction model was designed to reflect the arrival validation and DML activities of each of the six previously identified SIDPERS transactions. Figure A1.3-1 identifies the major functional areas of this model. Each functional area is represented by an IPSS service or services within the model. The function of each of these services are defined as follows:

System Loading

Within the IPSS model, the arrival of a transaction at a work station is represented by an exogenous event (defined within the Exogenous Event Stream Component). This event was scheduled to occur every 5.89 minutes plus or minus 5 minutes. This is based upon the following assumptions and parameters. First, the data base represents 203 soldiers; second, that there are an average of 8 transactions per month per soldier; third, that data base updating activities through

terminal interaction is possible 8 hours per day, 20 days per month. These figures were used to derive the transaction arrival event frequency of 5.89 minutes. The deviation of 5 minutes was selected arbitrarily to represent variation in the arrivals. In order to model a larger data base or different arrival patterns, a single statement need be changed in the procedure associated with the event. This fact is documented within the procedure and the statement identified for ease of modeler experimentation.

Transaction Arrival and Sequencing

This service represents the arrival of a transaction at a work station. The model assumes a single terminal and hence the transaction may be queued if the previous one has not completed. The model also assumes a FIFO (first-in/first-out) queueing discipline. The time that a transaction waits in this queue is a statistic automatically provided by IPSS. When a transaction is selected for processing, it is removed from the queue, the transaction Selection Service is invoked, and the service waits its completion. The total elapsed time from transaction selection to completion is also maintained by IPSS. Upon completion, the next transaction is selected from the queue and the service becomes idle. Idle time and busy period statistics are also kept. All the statistics mentioned above are maintained for each of the services and will not be further documented in this section.

This service can easily be modified to reflect an arbitrary number of terminals simply by establishing the maximum number of terminals permitted to be in operation at once, and keeping track of the current number. Only those transactions which arrive when the current number is equal to the maximum number need by queued. An infinite number of terminals can be modeled simply by not comparing these two variables and hence never queueing a transaction.

Transaction Selection

This service selects the type of transaction that has arrived and associates with it a set of parameters.[2] The selection is made by a probability distribution function based on relative frequency data provided by the Schaaff (SCH77) study. Once this selection is made, this service invokes the terminal interaction and validation service and waits for its completion. When the processing associated with this lower level service has completed, the Transaction Selection Service, determines if the transaction has been canceled (via a return code) and, if not, invokes the proper Application Service and waits for its completion.

Terminal Interaction and Validation

This service represents all the activities associated with the entry and validation of a transaction by a terminal operator; and can be conceptually divided into two parts, namely those activities that occur in obtaining valid transaction through terminal interaction in direct mode and those activities required to process and complete the transaction in tutorial mode. These processes are both characterized through tabular data and probability distribution functions. Input from the terminal, for example, is represented by advancing the simulator clock. In separate models, facilities representing the hardware and software processes are characterized and thus these activities are more accurately measured. The tutorial input mode requires several more parameters than the direct mode to characterize the events. The final activity of this service is to determine whether the transaction is to be canceled. This is communicated to the invoking service by a parameter value.

---

[2] These parameters represent the transaction characteristics and are derived from the data in Tables A1.2-1 and A1.2-2.

Application Services

The model contains a set of six functionally similar services, each of which represents the processing performed by an application program for one of the transactions. These services are all invoked by the Transaction Selection Service and they all at some time invoke the IDMS Interface Service. The processing performed by these transactions is represented by a series of DML invocations to this lower level service. The internal data manipulation activities are not represented in the model since their impact on performance is assumed to be minimal. The DMLs are identifical in type and sequencing to those reported by Schaaf (SCH77); their format is further discussed in the next section.

While not all performance evaluations characterize application processing to the detail represented in this model, this approach was deemed best for this model for the following reasons: (1) the data was readily available, (2) the mechanics of model construction was facilitated by IPSS language modeling features and (3) the model architecture emulates the processing performed by the real system. Thus, when the model is used in the future it will be evident what processes are being modeled.

Within IPSS, a service is dynamic facility and as such statistics are automatically collected for its use. For each of the services identified in Figure A1.3-1, utilization statistics are produced. These include: busy period, idle period, concurrency, seizure, and transaction transmit time statistics. In addition, queueing statistics are also produced. However, because of the nature of the model, only the Transaction Sequencing Service has queueing associated with it. These queueing statistics include: busy period, idle period, queue length, queue entry, and transaction transmit time statistics.

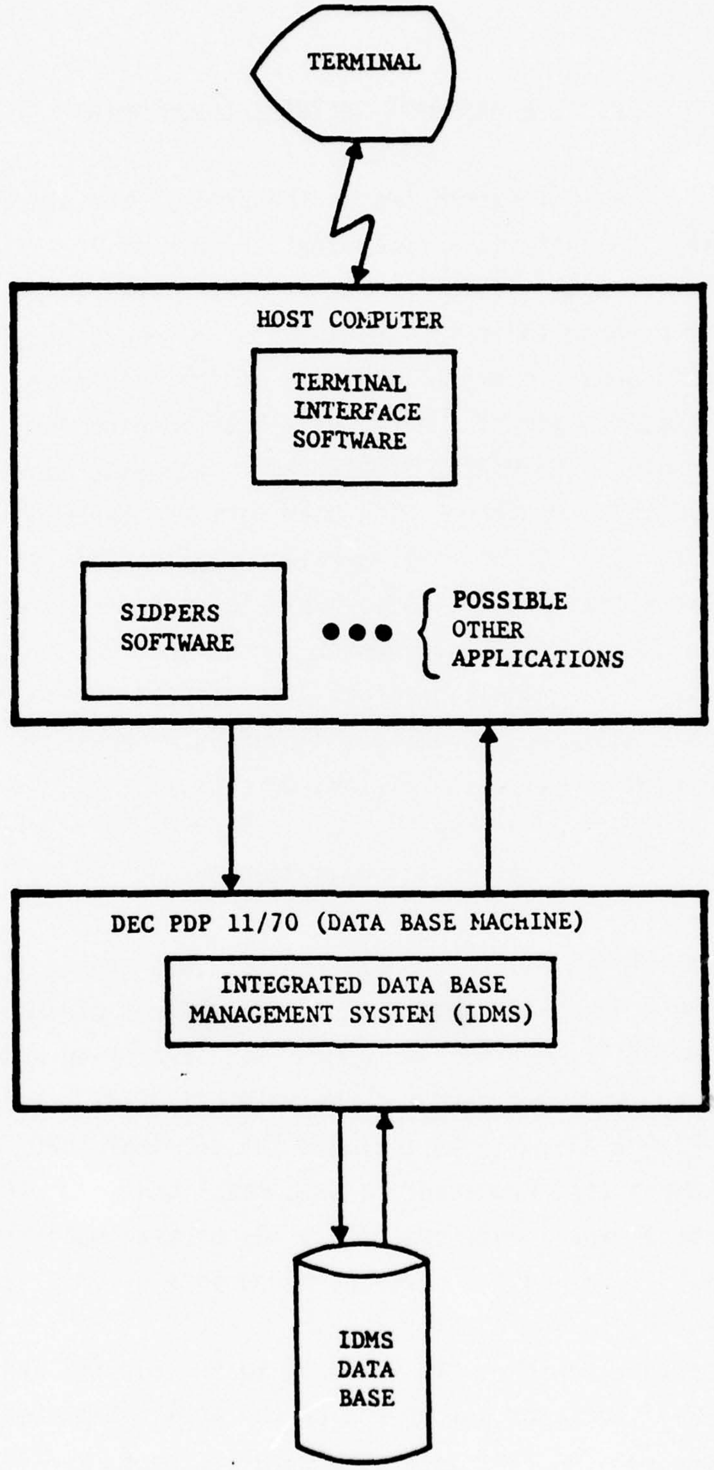


Figure A2-1 Experimental Version of VIABLE SIDPERS Processing Environment

A2. THE HARDWARE/SOFTWARE ENVIRONMENT

The configuration established as the protype for the VIABLE SIDPERS project employs the host/backend concept (Figure A2-1). The IDMS SIDPERS data base would reside on and be controlled by the backend machine. The purpose for this structure is to reduce the load on the host machine in order to extend the life of the mainframe. Previous studies have not designated a particular host machine for use by USACSC. Currently, the VIABLE SIDPERS test data base is maintained on the USACSC HCC IBM 370 system, with both host and backend machines residing there. One of the most immediate applications of the models developed during this project is to evaluate different host environments, in particular, the IBM 370 environment versus the PDP-11/70 environment. Thus, the goal of the modeling effort with respect to the host machine is to develop a model of a machine independent generalized host machine, allowing for each substitution of environments.

A2.1 The Host Machine

To accurately model the IBM 370/165 host was deemed to be too time consuming a task with regard to the objectives of the project. Therefore, an idealized host environment was identified which paralleled the hardware and software needs of the VIABLE/SIDPERS. Host hardware is shown in Figure A2.1-1. It included the terminal, CPU, secondary storage equipment (I/O Processor, disk control unit and eight IBM 3330 type disk units), and a data channel to the backend machine. Also shown in this figure are the hardware components assumed for the backend computer.

Host software functions in addition to the SIDPERS are shown in Figure A2.1-2. The major components of the application/host environment are the teleprocessing (TP) interface with the user, the TP interface with the backend machine, the host's disk I/O software, and the SIDPERS

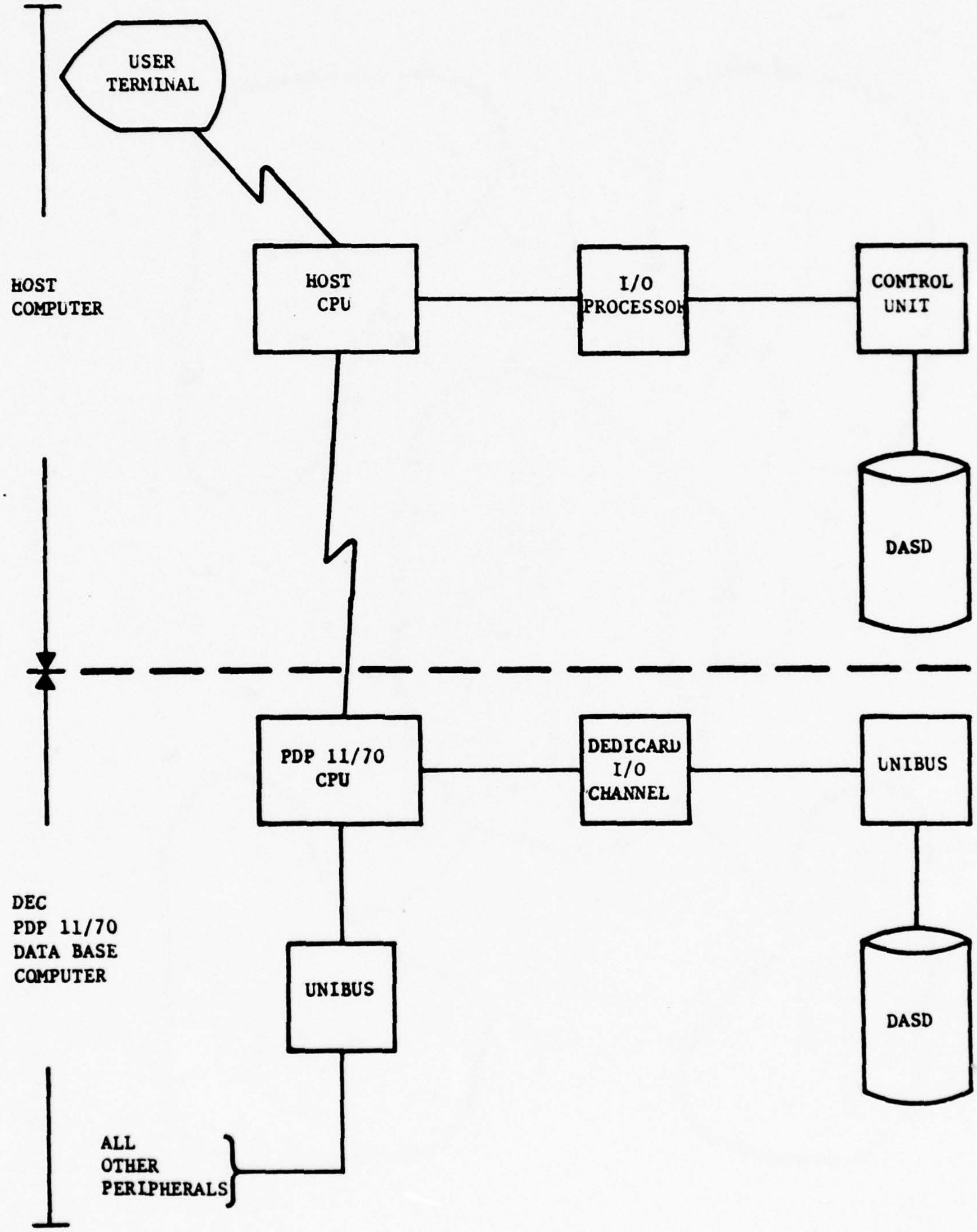


Figure A2.1-1 Modeled Hardware Configuration

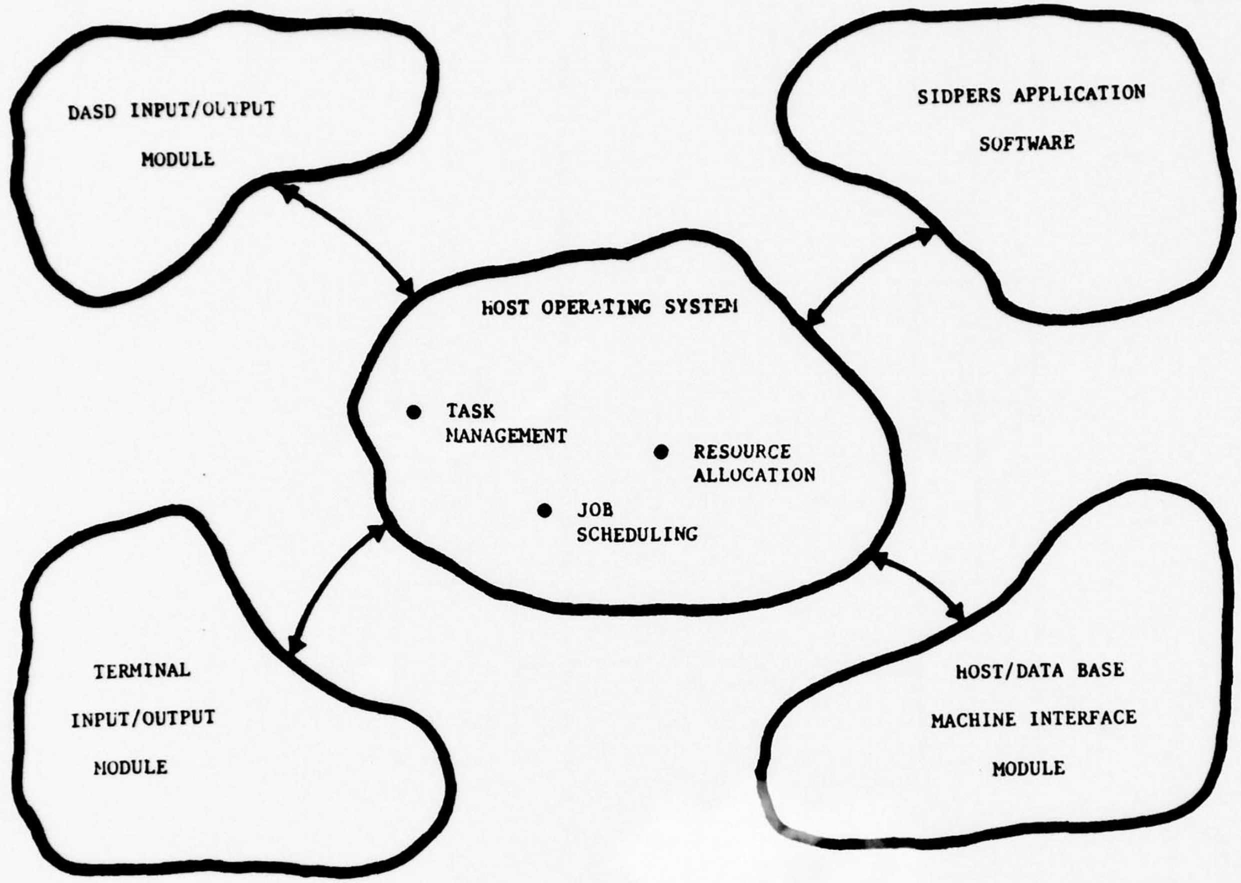


Figure A2.1-2 Modeled Host System Resident Software

application itself. The generalized host operating system (OS) is assumed to be reactive in operation, i.e., executing only to respond to the needs of the other software components.

Since VIABLE SIDPERS is interactive, communication with the user is maintained throughout each step in the process. In this capacity, the host performs the function of scheduling the communication, managing the TP line and sending/receiving the messages. After the initial log-on procedure the host operating system schedules the initiation of SIDPERS processing. The input and validation of SIDPERS data involves, in addition to the host's TP functions, disk I/O for the purpose of retrieving error messages for transmission to the terminal.

The crucial element of this model is the liaison between the application in the host machine and the DBMS in the backend machine. The DML request must be transmitted to the backend for processing and the retrieved record must be received. To accomplish this, the host supports an interface subsystem which formats the DML request, reformats the resultant record, handles the asychronous transmission and reception of messages to and from the backend, and controls the necessary devices for this communication link. It is assumed that each of the host modules rely on the basic operating system functions of task management and resource allocation. As in the actual system, the performance of these host functions is transparent to the processing of the application.

In order to make the model of the host environment machine independent, several conventions were employed. When scheduling was required (job, disk, or I/O), a FIFO queue was assumed for that host module. Resource allocation algorithms for TP and IOCS needs were also assumed to be FIFO. Any algorithm, however, can be substituted by changing only the host scheduling services without effecting the model's application services.

A2.2 The Backend Machine

The particular computer system selected by USACSC for the backend machine is the PDP-11/70. As a result of this prior decision a more

detailed and specific model could be built. The hardware configuration for the existing USACSC AIRMICS PDP-11/70 was modeled. Secondary storage characterization conformed to the IDMS a fixed page environment, i.e., the physical data records of the data base with a page size of 1,024 bytes and a direct file organization. Figure A2.2-1 shows the hardware configuration modeled.

The crucial aspect of modeling the backend machine for this project was to identify and characterize the interactions between the PDP-11/70 operating system and the IDMS applications. The main application oriented functions to be coordinated by the resident PDP-11/70 OS are illustrated in Figure A2.2-2. The internal functions of the backend machine are essentially the same as the generalized host except for UNIBUS management. The concept of the UNIBUS (a bidirectional data path through which all resources, including CP and memory, communicate) is central to PDP-11/70 processing. But its effect on processing is transparent to the application. The backend PDP-11/70 has an interface in the host. Because the PDP-11/70 is not dedicated to data base applications, and because USACSC has plans to support more than one data base per backend machine, each invocation of IDMS is treated as a separate subtask. The problem of concurrent updating by multiple users, however, forces the need for a single entry point or interface, called CAMP (Central Access Monitor Program). Even though entry to IDMS is controlled via the application, the actual operation of IDMS necessitates many requests for task management, resource allocation and I/O control. The model of the PDP-11/70 operates, as was assumed for the model of the host machine, in a reactive mode.

The object of the IDMS processing, retrieval of the specified physical record, necessitated the scheduling and use of I/O hardware which comes under the preview of the PDP-11/70 analog of the host's disk I/O module. Passing retrieved records to the host requires the aid of the OS to handle the sequencing of completed IDMS tasks and de-allocation of resources.

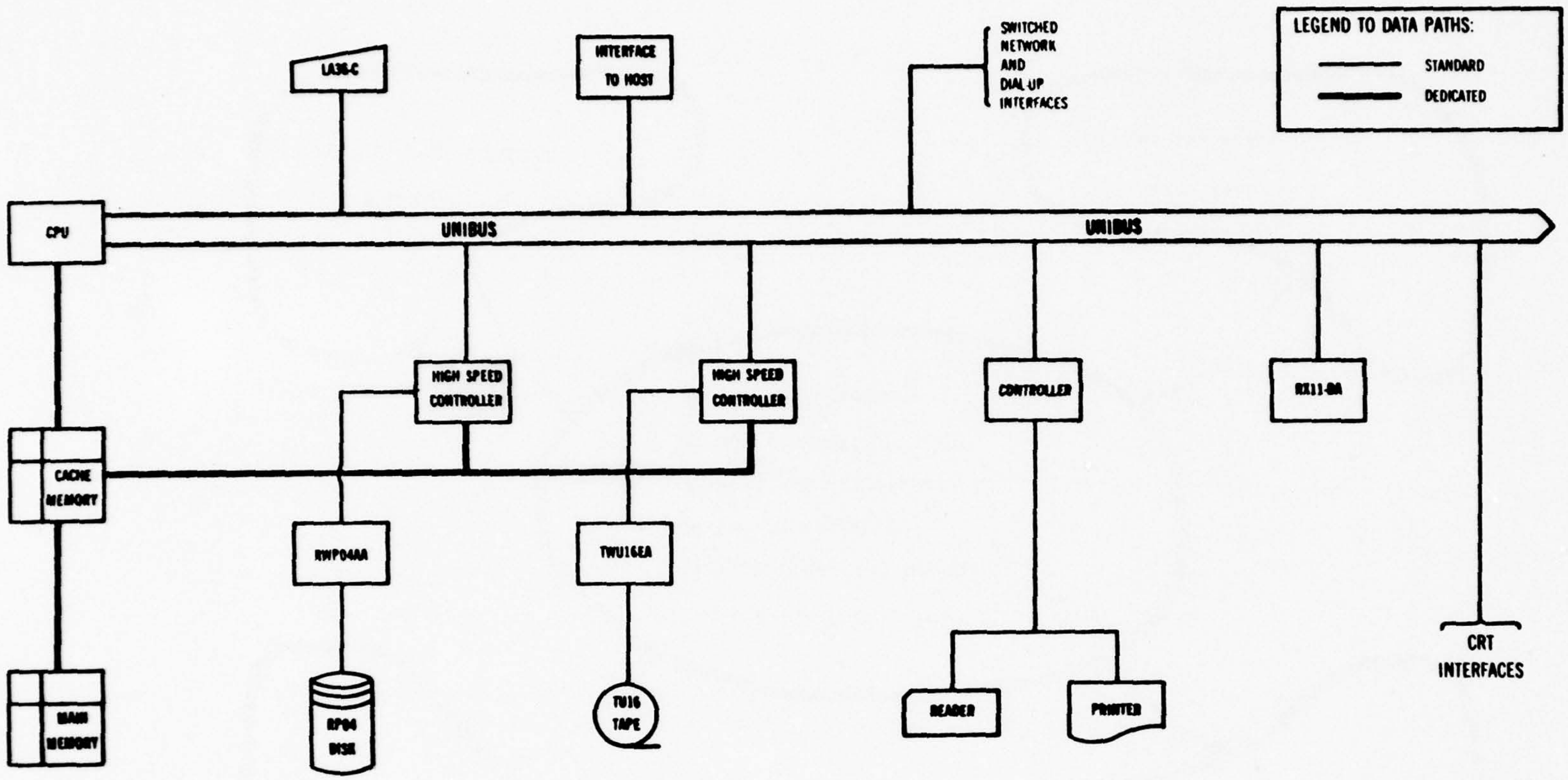


Figure A2.2-1 USACSC PDP 11/70 Configuration

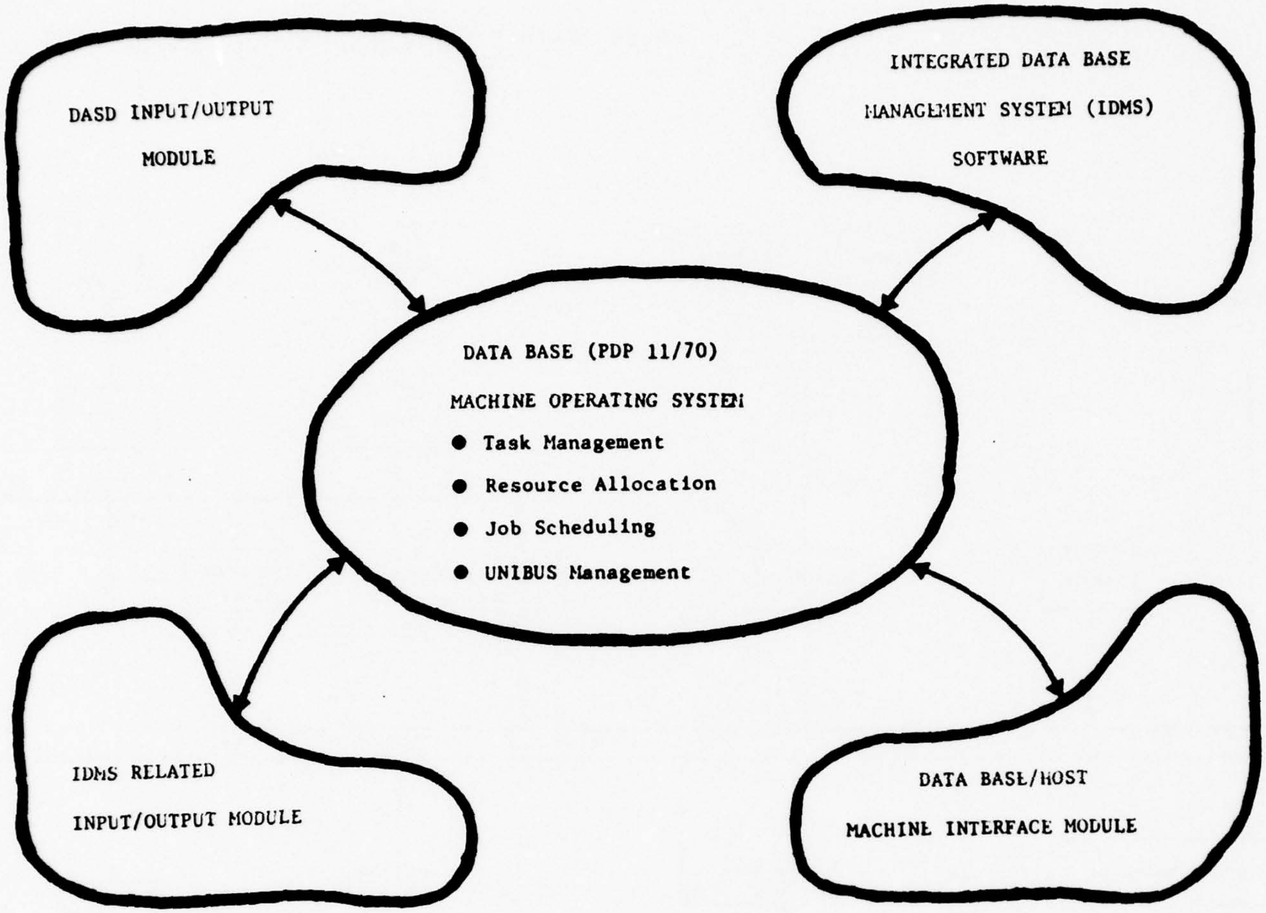


Figure A2.2-2 Modeled Data Base System Resident Software

The job and task scheduling disciplines employed by PDP-11/70 OS are priority queues based on the immediacy of use of the UNIBUS and were so modeled. All tasks performed by the PDP-11/70 priority scheme resulted in, essentially, FIFO queues.

A2.3 Integrated Database Management System - IDMS

The Integrated Database Management System is a network oriented implementation of a subset of the 1971 CODASYL Database Task Group specifications. The following activities were involved in making the conversion from sequential, batch processing to the IDMS on-line integrated database. First, the architecture of the integrated structure of the database was determined. Each of the files identified in Figure A2.3-1 became (in IDMS terminology) "record types" whose interconnections are represented by "sets". Second, the COBOL application programs for the six transaction types was rewritten to reflect IDMS processing. The major impact was in the incorporation of a centralized description of the database and in changing the I/O statements to standard IDMS Data Manipulation Language (DML) commands.

IDMS performs the following functions in response to an application DML. First it validates the request using the sub-schema definitions. Next it translates the request into a virtual page reference, determines if the page is present in its internal buffer, and requests the page from the underlying file management system when it is not already present. Finally, it places the subschema record into the application's work area and returns control to the application.

IDMS has its origins at the B.F. Goodrich Company where it was initially developed. The Cullinane Corporation (Wellesley, MA) now maintains product responsibility. IDMS operates on IBM 360/370 hardware. In addition, it is available through the Digital Equipment Corporation as DBMS-11 which operates on PDP-11/70 hardware.

The following paragraphs briefly introduce the major IDMS system components which were incorporated into the IPSS model (CUL76):

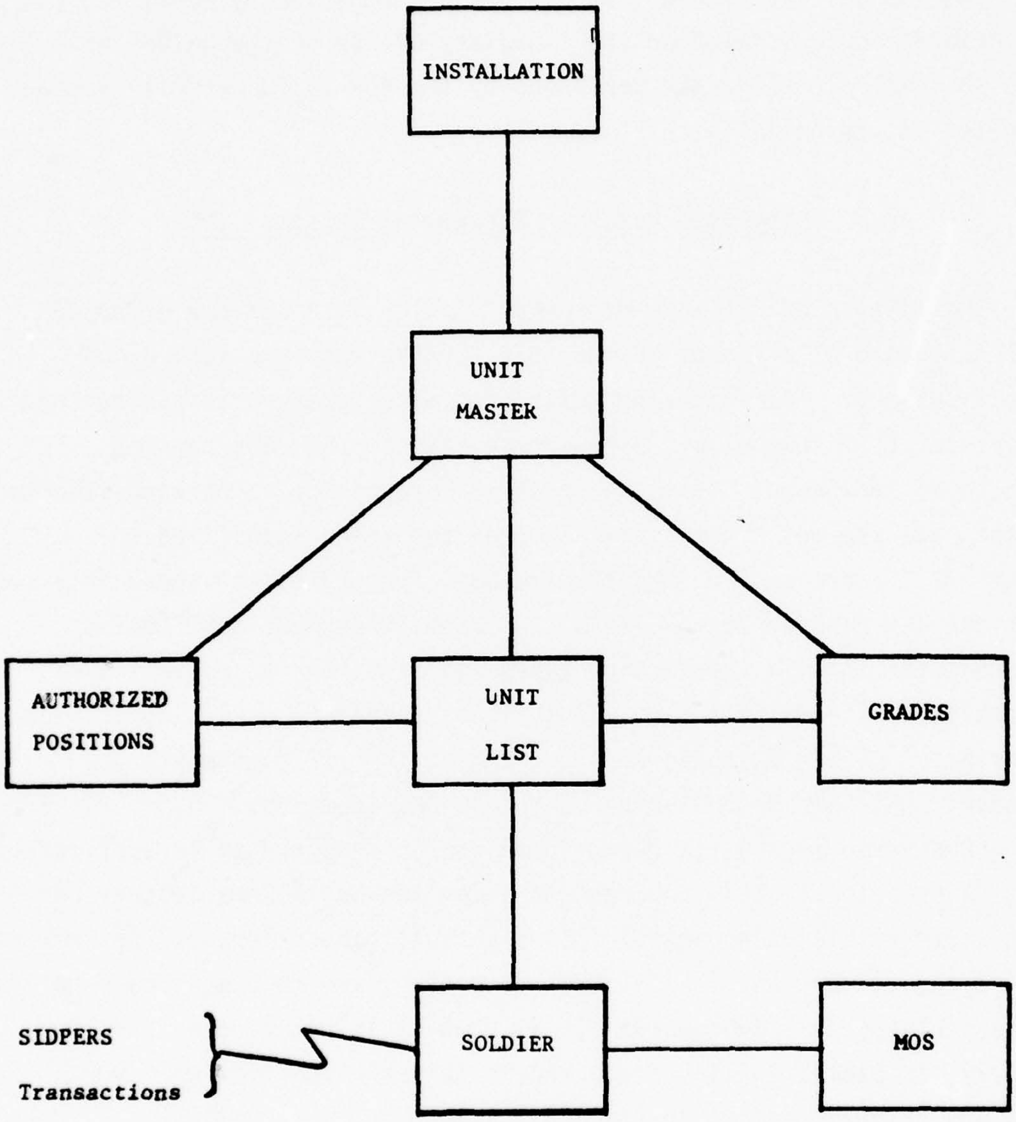


Figure A2.3-1 Modeled IDMS Schema for SIDPERS Data Base

Data Description Language (DDL) -- A facility for describing data independent of the programs that process it. Within IDMS the DDL consists of three parts:

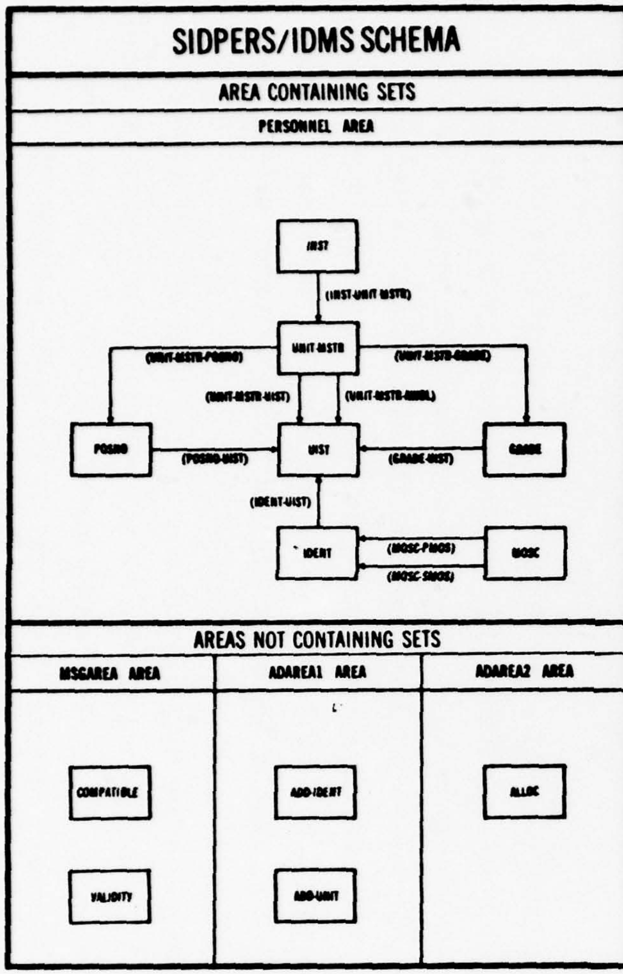
1. Schema DDL -- provides a complete description of the database including names and attributes of all data items, records, sets, and areas.
2. Device Media Control Language (DMCL) -- provides control over all the disk files which are to be on-line.
3. Subschema DDL -- provides a description of those data items, records, sets and areas to be used by one or more application programs.

Data Manipulation Language (DML) -- A facility for accessing base. It provides a wide range of commands for requesting database services which store data in the database or returns data to the user's program.

Schema Description for SIDPERS

The SIDPERS test database was designed to support the six transaction types identified earlier. The overall system architecture reflects twelve record types organized into four IDMS areas. These areas reflect functional differences in the data that is stored and its use. PERSONNEL area contains the individual solder and unit records of an installation. The MSCAREA contains diagnostic and error messages. The ADAREAL and ADAREA2 Areas contain additional personnel data. The overall structure is shown in Figure A2.3-2.

The record types belonging to each of the areas are identified in Table A2.3-1. The record occurrence and record size attributes of each of the record types are also identified in this table. The personnel area is of primary interest since most of the SIDPERS transaction activity references it record types and sets. This area consists of 3,909 records distributed among 7 record types. Although the GRADE records account for 68% of the total, the IDENT records (5.2% of the



RECORD TYPE AND SET CHARACTERISTICS

SET OCCURRENCE CHARACTERISTICS

SET NAME	SET RECORD TYPE		PROBABILITY OF SET OCCURRENCE GIVEN OWNER RECORD OCCURRENCE	AVERAGE NUMBER OF MEMBER RECORDS PER SET OCCURRENCE
	OWNER	MEMBER		
GRADE-LIST	GRADE	LIST	.94	2
IDENT-LIST	IDEN	LIST	.86	2
INST-UNIT-MSTR	INST	UNIT-MSTR	1.00	63
MOSC-PMOS	MOSC	IDEN	.50	3
MOSC-SMOS	MOSC	IDEN	.42	2
POSNO-LIST	POSNO	LIST	0.00	0
UNIT-MSTR-ANNL	UNIT-MSTR	UIST	0.00	0
UNIT-MSTR-GRADE	UNIT-MSTR	LIST	1.00	32
UNIT-MSTR-POSNO	UNIT-MSTR	POSNO	.90	2
UNIT-MSTR-LIST	UNIT-MSTR	UIST	1.00	2

RECORD TYPE VOLUMES

AREAS NOT CONTAINING SETS		AREA CONTAINING SETS	
AREA-Name Record Type Name	Volume of Records	Area-Name	Volume of Records
ADAREA 1		PERSONNEL	
ADD-IDENT	200	GRADE	2,656
ADD-UNIT	85	IDEN	203
		INST	1
ADAREA 2		MOSC	95
ALLOC	95	POSNO	403
MSGAREA		ULIST	468
COMPATIBLE	155	UNIT-MSTR	63
VALIDITY	95		

Figure A2.3-2 SIDPERS Data Diagram

Table A2.3-1 Record Occurrence Characteristics

Record Type Within Area	Record Occurrence		Record Size	
	Number	% of Total (within area)	Length* (Bytes)	Percent Used of Space Available Within Area
PERSONNEL AREA				
INST	1	0.0	48	.01
UNIT-MSTR	83	2.1	52	.79
IDENT	203	5.2	120	4.12
UIST	468	12.0	84	6.82
GRADE	2,656	68.0	20	11.78
POSNO	403	10.3	140	9.45
MOSE	95	2.4	236	3.67
ADAREAL AREA				
ADD-IDENT	200	70.2	284	23.13
ADD-UNIT	85	29.8	552	18.85
MSGAREA AREA				
COMPATIBLE	155	62.0	296	14.93
VALIDITY	95	38.0	512	15.65
ADAREA2 AREA				
AALOC	95	100.0	96	.15

*Includes IDMS Overhead of 8 bytes per record.

total) are most frequently accessed. Most of the transactions are oriented toward updating an individual soldier's data and thus the most common entry point is the IDENT record. This is accessed via the CALC set using the soldier's last name.

Figure A2.3-2 identified the organization of the personnel area record types into sets. Ten distinct set types (excluding the CALC sets) are employed to provide the inter-record traversal capabilities required by the SIDPERS transactions. Table A2.3-2 identifies these sets and displays their characteristics. The sets are identified by both an "external" name which is the set name used in the SIDPERS subset, and also by an "internal" name which is the set identifier within the IPSS models. For modeling purposes, two data items per set type were derived from analysis of the sample data base. These were:

1. the probability of an owner record type having a non-null set of member records associated with it (i.e., the probability of a set occurrence), and
2. the number of member records in a set occurrence for each set type.

Table A2.3-2 also identifies these data items for each set of the PERSONNEL AREA. Although the table shows the average number of member records, for modeling purposes probability distribution functions were used to more accurately characterize the database.

The records of the SIDPERS data base reside within specially formatted IDMS pages. These IDMS pages are virtual pages since they represent the data as known by IDMS and as such need not be reflective of secondary storage record organization. The page size itself is a function of the IDMS processing environment. Interactive SIDPERS with IDMS operating on IBM 370/165 equipment uses a page size of 3,156 bytes, whereas the page size for IDMS operating on a PDP-11/70 is projected to be 1,024 bytes. The test version of the SIDPERS data base has been loaded onto IDMS pages as shown in Table A2.3-2. The Personnel Area occupies 200 pages and is 36.6% full. The IPSS simulation model of this data base assumes a PDP-11/70 environment as the number of pages and page range was adjusted to keep the percent utilization the same while changing the page size to 1,024 bytes. This data is also reflected in Table A2.3-3.

Table A2.3-2 Set Occurrence Characteristics

Set Name External ID	Internal	Record Type Participation Owner	Member	Probability of Set Occurrence Given Owner Record	Average Number of Member Records
INST-UNIT-MSTR	INUM	INST	UNMA	1.0	83.0
UNIT-MSTR-POSNO	UMPOS	UNMA	POSN	.9	5.7
UNIT-MSTR-UIST	UMUI	UNMA	UIST	1.0	2.6
UNIT-MSTR-AWOL	UMAW	UNMA	UIST	0.0	0.0
UNIT-MSTR-GRADE	UMGR	UNMA	GRADE	1.0	32.0
POSNO-UIST	POUI	POSN	UIST	0.0	0.0
GRADE-UIST	GRUI	GRADE	UIST	.94	1.6
IDENT-UIST	IDUI	IDEN	UIST	1.0	3.0
MOSC-PMOS	PMID	MOSC	IDEN	.86	1.0
MOSC-SMOS	SMID	MOSC	IDEN	.57	2.0

Table A2.3-3 Page Utilization Within the SIDPERS Areas

Area	Page Size = 3,156 Bytes			Page Size = 1,024 Bytes	
	Number of Pages	Page Range	% Utilized	Number of Pages	Page Range
Personnel	200	1-200	36.64	617	1-617
ADAREA1	80	201-280	41.98	246	618-863
MSGA2EA	100	301-400	30.58	308	925-1233
ADAREA2	20	281-300	.15	60	864-924

DML Description for SIDPERS

A previous study of a VIABLE transaction has specified the number and type of IDMS DMLs that are issued as well as their sequence within the transaction (SHA77). These DML have been classified by type as shown in Table A2.3-4.

The function of each of these DML Commands is as follows:

FIND: Locates the selected record in the data base and returns its identity to the application program.

OBTAIN: Locates the selected record in the data base and returns the record as well as its identity to the application program.

MODIFY: Causes the specified record to be updated.

STORE: Causes the specified record to be stored in the data base.

ERASE: Causes the specified record to be physically removed from the data base.

CONNECT: Causes the specified record to be inserted into an identified set.

DISCONNECT: Causes the specified record to be removed from the identified set.

Table A2.3-4 SIDPERS IDMS DML Types

Category	DML
FIND/OBTAIN	
	FIND record-name RECORD FIND NEXT record-name RECORD of set-name SET FIND OWNER RECORD OF set-name SET
	OBTAIN record-name RECORD OBTAIN NEXT record-name RECORD of set-name SET OBTAIN OWNER RECORD OF set-name SET
MODIFY/STORE/ERASE	
	MODIFY record-name RECORD STORE record-name RECORD ERASE record-name
CONNECT/DISCONNECT	
	CONNECT record-name TO set-name DISCONNECT record-name FROM set-name

Within the IPSS model of VIABLE SIDPERS, each IDMS DML is represented by statements which invoke a lower level service and wait for its completion. Four parameters that are required to pass the DML information. These are:

1. DML Type (FIND, OBTAIN, etc.);
2. Qualifier (NEXT, OWNER, etc.);
3. Record Type Identified (ADDI, IDEN, etc.); and
4. Set Type Identifier (GALC, GRUI, etc.).

A3. MODELING USING THE IPSS DESIGN FACILITY

A3.1 A Functional View of the Modeling Design

The purpose of the host and the backend submodels is to provide specific computer system environments for the SIDPERS application and IDMS submodels. The relationship between these submodels can be seen in Figure A3.1-1. The SIDPERS submodel was constructed by characterizing the functional SIDPERS processes required by a SIDPERS transaction from the time it enters the host system until the completion of processing. These functional processes included such activities as input validation, backend interaction, and diagnostics transmissions. The SIDPERS submodel characterized these processes, however, in a computer system free environment, concerning itself only with the impact of these various processes on SIDPERS processing. Similarily, the IDMS submodel was constructed with no assumed computer system environment. Functions such as buffer and page management, DML invocation and host interfacing were characterized as part of the IDMS submodel, at the level of complexity of the IDMS request. Both the host and backend computing environments were treated as black boxes in these models. Thus, the initial preliminary model had the SIDPERS submodel interacting with the IDMS submodels (arrows A & B).

The purpose of the host and backend submodels were to provide a specific computer environment, including hardware, software and data elements, within which the SIDPERS and IDMS submodels could operate. The operating system functions that had been assumed to be operational in the application oriented submodels, but had been aggregated, were now characterized. For example, transmitting SIDPERS diagnostic messages to the user at the terminal involves accessing the message from the host's disks (including I/O scheduling, disk allocation and I/O operations) and the actual transmission (which requires TP scheduling, TP line allocation and I/O operations). With respect to the IDMS submodel, the function of page replacement, for example, involves task scheduling, resource allocation, and memory management. It is the characterization of these support functions that the host and backend submodels provided.

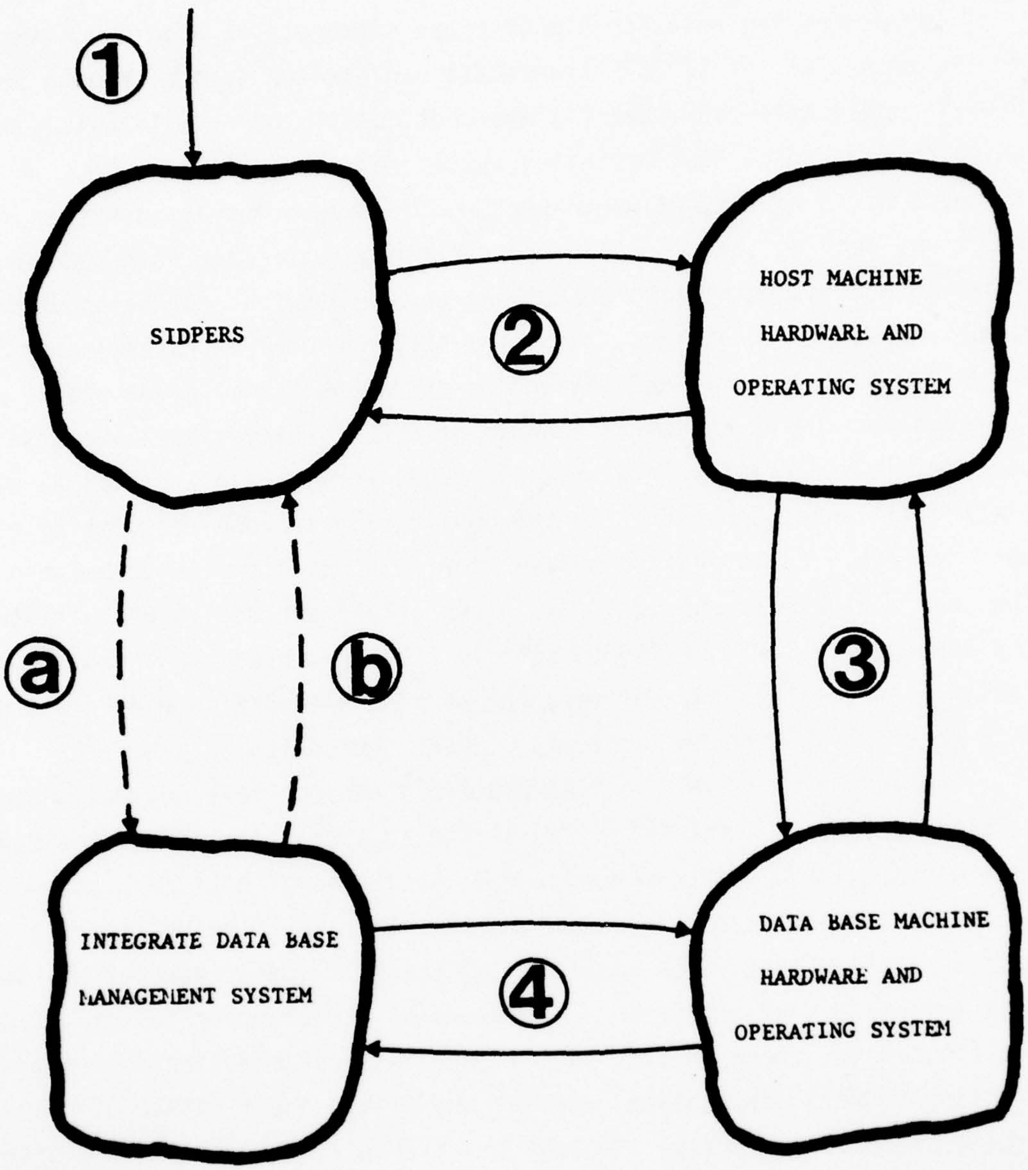


Figure A3.1-1 Relation of Submodel to Total Model

The processing relationship of these submodels is also demonstrated in Figure A2.1-1. A SIDPERS transaction enters the system through the SIDPERS application submodel (1) and continues to operate in this submodel until it requires a host operating system function. At that time it triggers off a host function to perform the needed operating system function, such as task, job or resource management (2). Both submodels continue processing other transactions and events that may be occurring simultaneously. In essence, then, the host submodel operates in parallel with the SIDPERS model, satisfying its operating system needs and coordinating the execution of concurrent SIDPERS processes. The host submodel is essentially in a reactive mode, residing in a continual wait state until awakened by the SIDPERS application submodel requesting a host service. Communication between the host is accomplished between the host and backend submodel (3). The backend submodel then initiates the appropriate IDMS service to satisfy the DML request sent from the host (4). From then on, however, the backend and IDMS submodels operate in a parallel as does the host and SIDPERS submodels.

Previous studies of the VIABLE/SIDPERS project have not designated a particular type of computer system as the host machine. Thus, the model that was developed has been designed to be a general host machine model, including those operating system functions that would be necessary on any machine that as host. The structure of the host model, however, is such that the particular software and hardware of any computer system can be substituted for those in the initial model without modification of the existing SIDPERS application submodel and little modification of the host submodel. Currently, the test bed SIDPERS/IDMS system resides on an IBM 370/165, both the host and the backend. Thus, although the structure of the host submodel in this project is general, the particular elements are specific to the USACSC HCC IBM 370/165. The backend machine, however, has been chosen to be a PDP-11/70 and the resultant backend model will reflect the operating system philosophies of this system.

In modeling these two types of environments, a similar approach was taken. It is assumed in the IPSS methodology that operating systems,

in general, perform certain types of services regardless of the particular machine. As Figure A3.1-2 illustrates, although an operating system may be servicing many different applications simultaneously, the coordinating and servicing of these applications is completed through a stage set of underlying operating system functions. The types of system functions performed are job scheduling, task management, resource allocation, I/O supervision and telecommunication control. Thus, both the host and the backend submodels were constructed by characterizing these functions in general and then particularizing them to the specific machine, IBM 370/165 and PDP-11/70. The structure of these submodels, however, is such that at any later time the hardware and/or software can be changed to represent another system.

It should be emphasized that these characterizations of the computer systems were not dependent on the applications to be run on them, i.e., SIDPERS and IDMS. A future application of these submodels is to change the loading on the computer submodels to include other, possible other SIDPERS, applications to determine the effect on SIDPER/IDMS processing. Thus, the scheduling and allocation policies that are coded for these submodels do not reflect the particulars of either SIDPER or IDMS and are, essentially, general task and resource managers.

A3.2 IPSS Model of Host Computer

The approach taken in using IPSS to model this computer system is to first model the static elements of the system, the hardware and the data and then model the dynamic software elements. The host computer that was modeled was the USACSC HCC IBM 370/165. To construct the hardware configuration of this machine, only those elements that were impacted by the SIDPERS application were modeled. Specifically, they were the secondary storage devices upon which the IDMS error message files were kept, the backend interfacing equipment, external terminals through which USACSC personnel invoke and interact with SIDPERS, and the connecting telecommunication equipment. Other hardware typical of an IBM

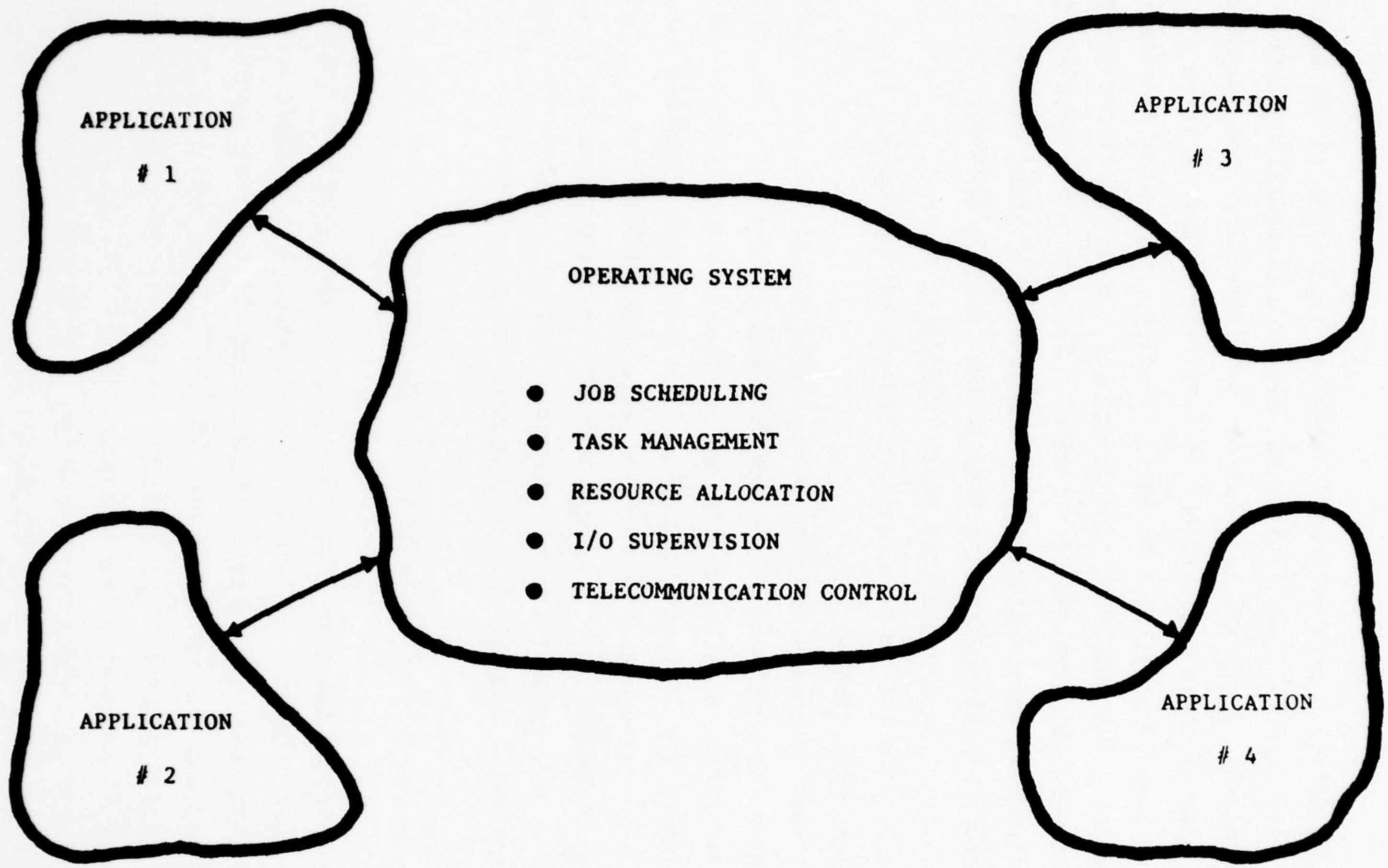


Figure A3.1-2 Modeled Operating System Functions

370/165 were not included as they would not be referenced in this model, but at a later point in time when other USACSC applications are added to the host model, additional hardware can be easily added.

Figure A3.2-1 illustrates the initial hardware configuration that was assumed for the host, IBM 370/165 computer. The particular secondary storage devices assumed were a bank of eight IBM 3330 DASDs, their associated control units, and single channel. A single CPU with 2 megabytes of main memory were assumed to characterize the host computer. The external terminals were assumed to be similar to IBM 2741 and the communication line between the host computer and these terminals was assumed to be a 2400 baud line. It should be noted that in an IPSS model the characteristics of anyone of these hardware devices can be modified without requiring the modification of the rest of the model. Thus, experiments evaluating different combinations of devices can be easily made.

The second step in constructing an IPSS model is to define the secondary storage data files. With respect to the SIDPERS application, the only data files associated with the host computer are the files of validity and compatibility error messages. The validity error file is accessed once for a compatability error by the SIDPERS application software. Although there was little data on the size and internal characteristics of these files the following was assumed: logical record length of 80 bytes, physical record length of 4000, (to fit into the VM OS page size of 4096), 1000 validity records, 200 compatibility records and placement of the two files on the same volume.

The crucial aspect of this and any IPSS model is the characterization of software processing. It has been mentioned that the operating systems of both the host and the backend computers were assumed to be in a reactive mode, responding to the needs of their respective applications. Thus, in order to determine the operating system functions that needed to be modeled in the host (and backend) IPSS submodels, the processing requirements of the SIDPERS (and IDMS) applications must be examined. The requirements for the host can be seen by examining the functional

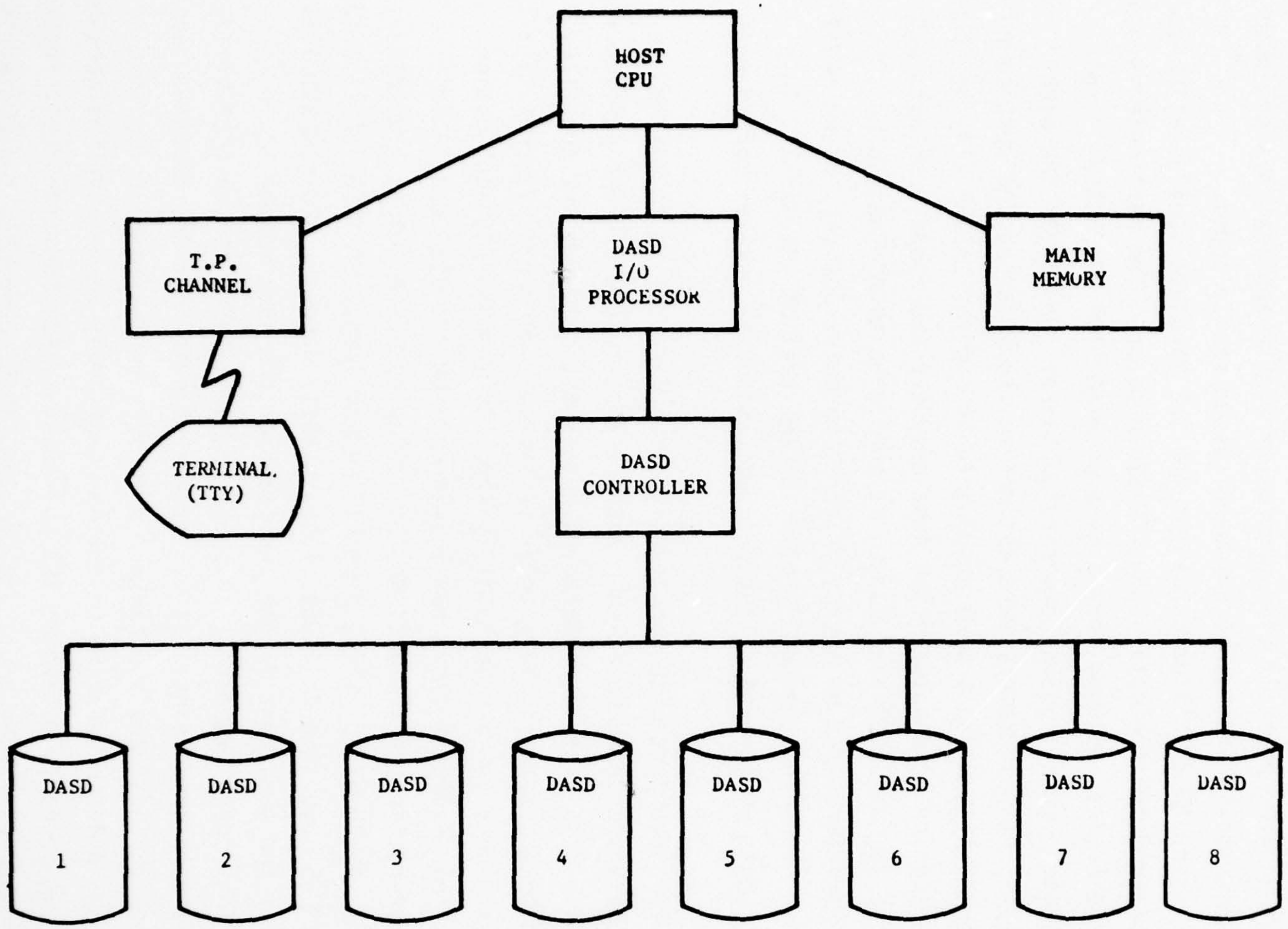


Figure A3.2-1 Assumed Host Hardware Configuration

steps that a SIDPERS transaction goes through that were detailed in Section A1. Summarizing, there were five basic steps as shown in Figure A3.2-2: log-on, SIDPERS scheduling, data input and validation, IDMS/backend interaction, and log-off. At each of these steps particular operating system functions are required at verying levels of complexity.

To satisfy these needs, certain salient operating system functions were identified. These were: the scheduling and transmission of data to/from the user, the scheduling of a SIDPERS application within the host, the communication requirements of interactive data input and the necessary resource allocation and I/O scheduling to access the error message files and the scheduling transmission and repetition of messages to/from the backend computer. Underlying all these processes is the synchronization and coordination of the SIDPERS application functions and the host operating system functions which may be executing in parallel, as provided by general task management and resource allocation.

The relationship between the IPSS services that characterized the SIDPERS application and the IPSS services that performed the host functions are shown in Figure A3.2-3. Being that SIDPERS is an interactive system, constant communication between the user and the SIDPERS system, through the host, must be maintained. Thus, each of the five SIDPERS application steps, in some way, require the scheduling and transmission of information to or from the user. In the resulting IPSS submodel this interface was handled by the service XTCAM, which acted as scheduler for TP to the user. It was awakened by a SIDPERS module and returned to the wait state. The requesting SIDPERS module was then in control of the use and disposition of the TP line. The second step was to specifically scheduled the SIDPERS application for execution in the host system. This was performed by the IPSS service JOBSCH which controlled the allocation of main memory and central processor and placed the incoming SIDPERS application in the ready queue. It is general, though, so that other non-SIDPERS jobs could also be scheduled.

The third SIDPERS application step, data input and validation, was characterized in the SIDPERS application submodel by the services VALID,

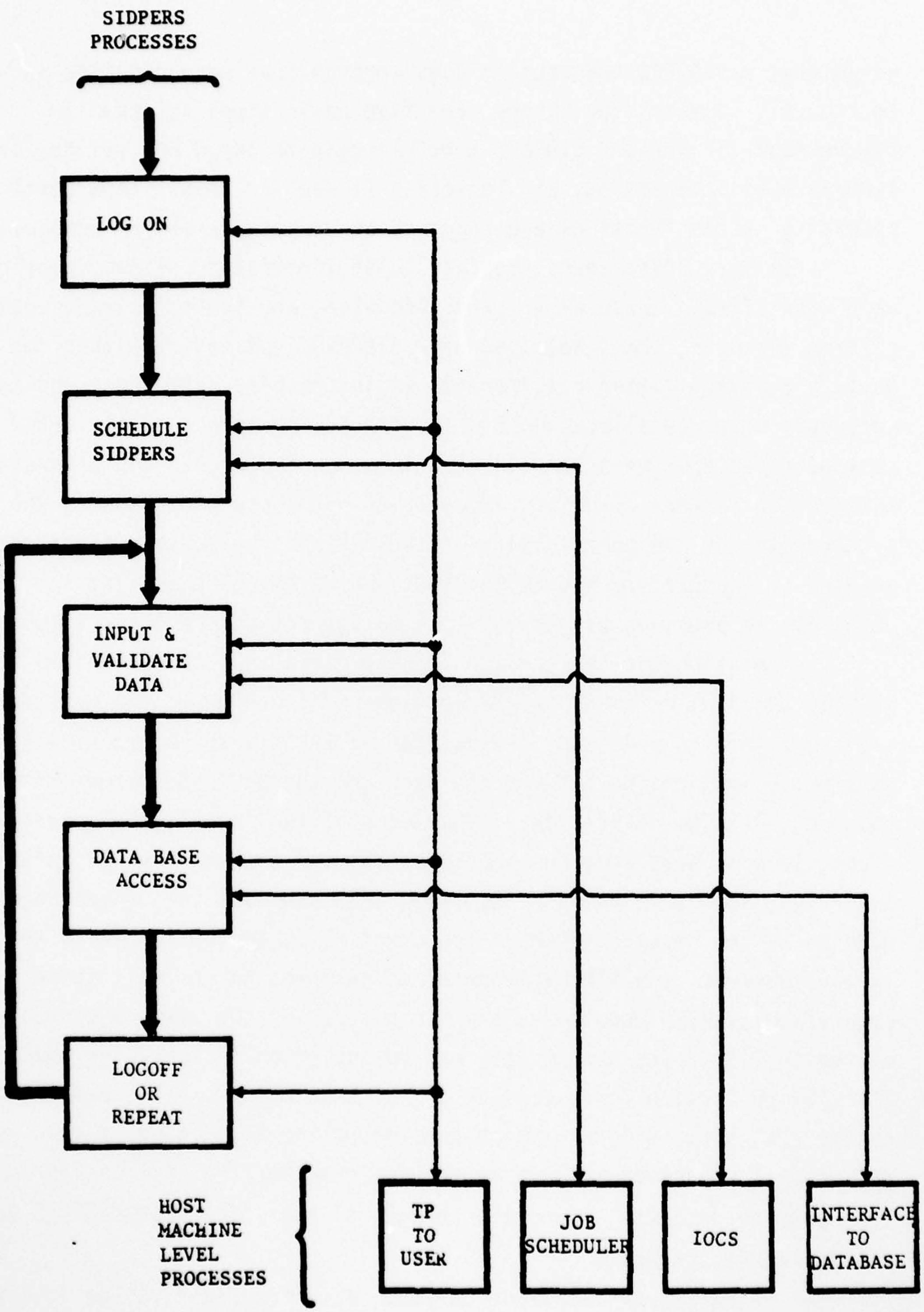


Figure A3.2-2 Relation of Host Machine Processes to SIDPERS Processes

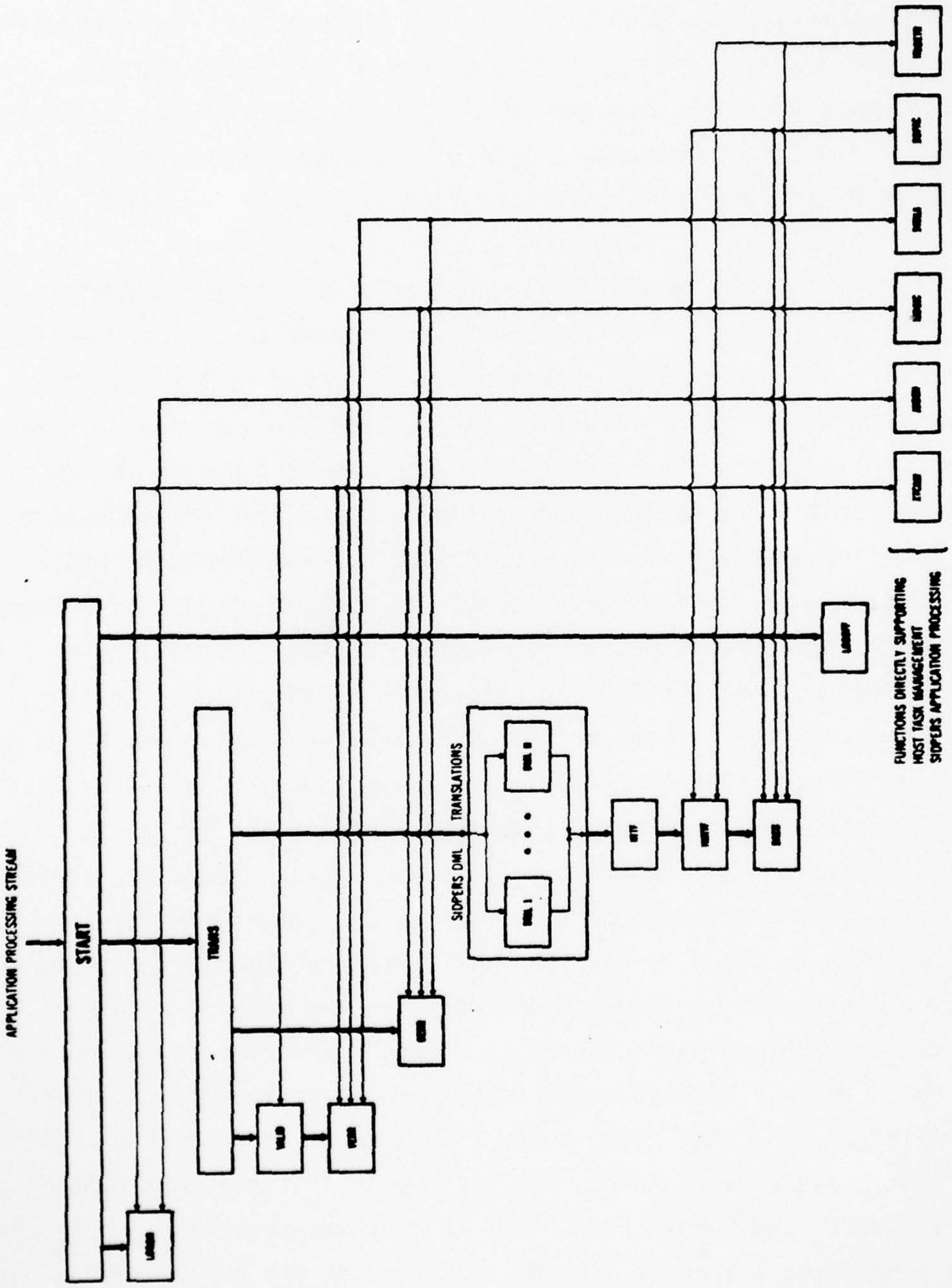
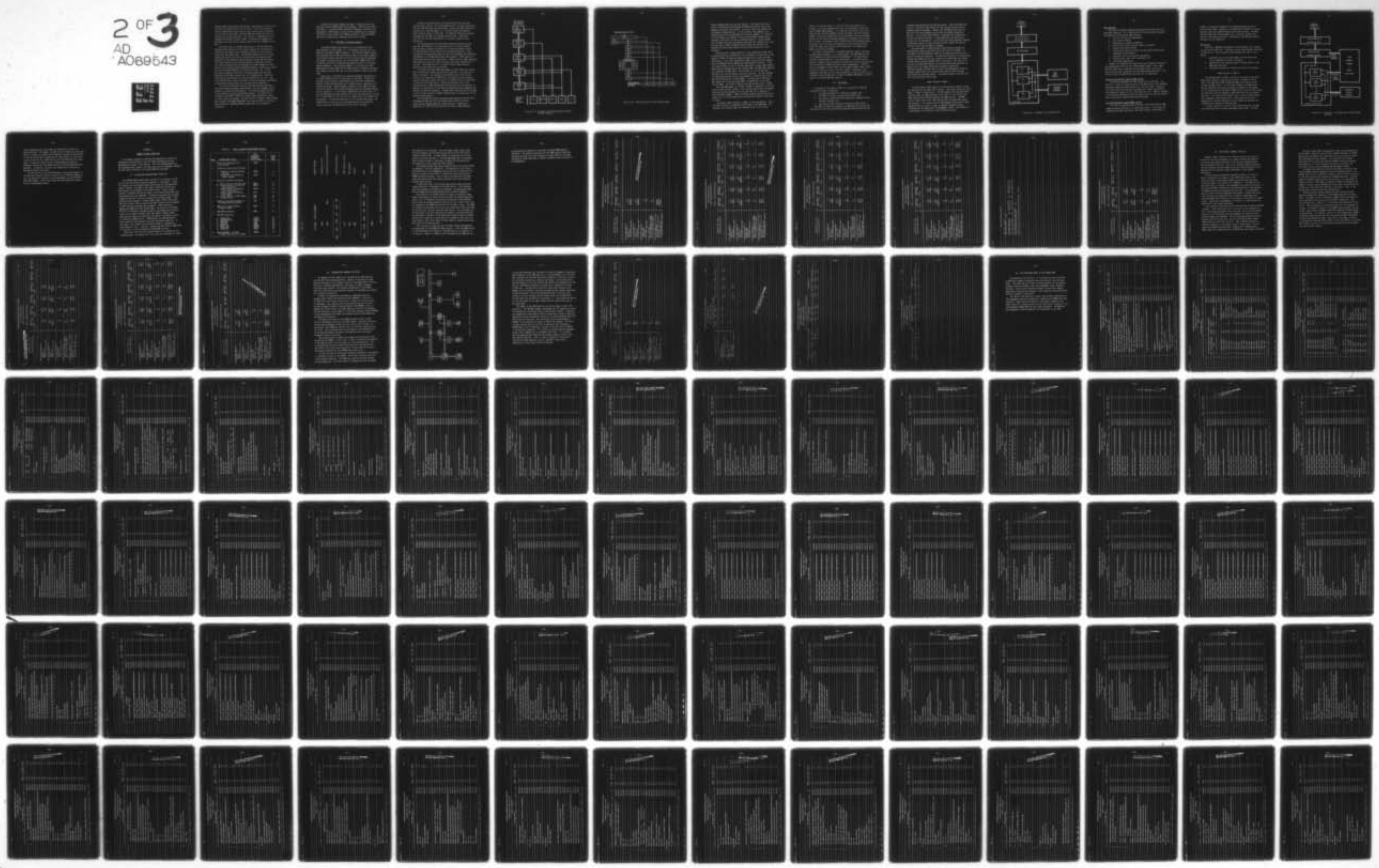


Figure A3.2-3 IPSS Realization of Host/SIDPERS Model

AD-A069 543 OHIO STATE UNIV RESEARCH FOUNDATION COLUMBUS F/G 9/4
A SIMULATION LANGUAGE FOR EVALUATING INFORMATION PROCESSING SYS--ETC(U)
SEP 78 T G DELUTIS DAAG29-77-G-0203
UNCLASSIFIED ARO-15356.1-A-M NL

2 OF 3
AD
A069543



VERR and CERR, representing, respectively, input process, validity error accessing and compatibility error accessing. Each of these modules involves communication with the user, thus, necessitating interaction with the host service XCAM. VERR and CERR also require the access of the host disk storage subsystem. This process is regulated by a disk scheduler and device allocator, XHDSK. The actual I/O commands are performed by an IOCS routine labeled DISKA in the IPSS submodel of the host.

The fourth step in a SIDPERS application is the DML processing, including the interface with the backend computer for the purposes of sending the DML request to the backend, and receiving the retrieved record from the backend IDMS. The SIDPERS submodel services for each transaction type and the service INTF determine the particular DML required for the transaction and perform any pre/post DML processing. The actual interface to the backend computer is performed by HINTF. This service characterizes the packaging of the DML request for transmission to the backend, buffer management (BUFALC), scheduling of the TP line to the backend (HOBETR) and allocation of that line. In the preliminary submodel of the host, the backend processing was black boxed and service BACK characterized this implementation. BACK also controlled the deallocation of resources (buffers and TP line). The final SIDPERS step of log-off parallels the log-on procedure, performing allocation (now deallocation) of resources and job scheduling.

In order to characterize the host operating systems functions as they would normally operate, asynchronously, the IPSS submodels had to operate (or simulate) asynchronously. The interface between the host and the backend is also an asynchronous process. The IPSS built-in statements of POST/WAIT EVENT allow XTCAM, XHDSK and HOBETR to remain in a wait state until needed. The IPSS built-in statements FIND IN QUEUE, WAIT IN QUEUE, and REMOVE FROM QUEUE allow these services to maintain a queue containing suspended SIDPERS transactions via any queueing discipline.

Testing of this host submodel was done in conjunction with the previously tested SIDPERS application submodel. Since the host model was always in a reactive mode, completely independent testing was not appropriate. Additionally, the use of this submodel was to determine the effect of host support for SIDPERS processes to that the SIDPERS application submodel was a necessary part of this testing.

A3.3 IPSS Model of Backend Computer

The backend computer was modeled by first characterizing the great features of that system, i.e., the hardware and data, and then modeling the dynamic software features. Since previous USACSC studies had indicated that the PDP-11/70 would be used as the backend computer, a more specific model than that built for the host could be developed. The hardware configuration for the existing USACSC AERMICS PDP-11/70 at Georgia Institute of Technology was modeled as part of the Systems Reserve Component. Since backend computer is to be used for other applications besides supporting the IDMS system, the complete configuration needed to be modified, not only those elements directly affecting IDMS.

Figure A2.2-1 illustrates the idealized hardware configuration of the USACSC AIRMICS PDP-11/70 system used in the model. Although the physical system itself was unavailable, sufficient DEC documentation existed so that detailed modeling of individual device characteristics could be achieved. Several hardware elements of this model that have significant impact on the software processing and thus software modeling, of this system are: (1) the central UNIBUS data and control path; (2) the two dedicated high speed controllers to disk and tape drives; (3) the use of cache memory by the core and; (4) communication link between host and backend. It should be noted that in this and many IPSS models, the characteristics of any one of the hardware devices can be modified without requiring the modification of the rest of the model.

Defining the characteristics of the particular data sets that reside on this system is the second modeling step. For this application, only the IDMS test data base was modeled. Each of the IDMS records was, essentially, a 1024 byte page (i.e., physical record) with varying highly logical records in each page. The IDMS test data base contained 200 pages and the IDMS file organization was direct. Other data sets for this and other applications can be easily added to these definitions.

The characterization of the backend software processing follows much the same pattern as the characterization of the host software. Both operating systems are assumed to be in reactive models, waiting for and responding to the needs of the resident applications. Thus, to determine the operating system functions to be modeled, the processing requirements of the IDMS applications were examined. A DML request in the backend follows a five step process: request arrival, IDMS scheduling, IDMS processing, physical record retrieval, and record transmission to the host. (See Figure A3.3-1). At each of these steps particular operating system functions are required.

To satisfy functional IDMS requirements for DML processing, certain PDP-11/70 operating system functions were identified. As Figure A3.3-2 shows, these were: the scheduling and transmission of data to/from the host, the scheduling of IDMS within the backend, and the necessary resource allocation, buffer allocation, I/O scheduling and I/O execution to retrieve the desired record. Underlying processes such as the synchronization of IDMS and backend functions, which may be executing in parallel, is provided for in the model by generalized task management and resource allocations procedures. Although CPU scheduling was not included as part of these functions, UNIBUS scheduling was an integral part of the synchronization process.

The relationships between the general DML processing step and their corresponding IDMS services, and between these services and those characterizing backend operating system functions is depicted in Figure A3.3-2. Both the first and the last steps in DML processing

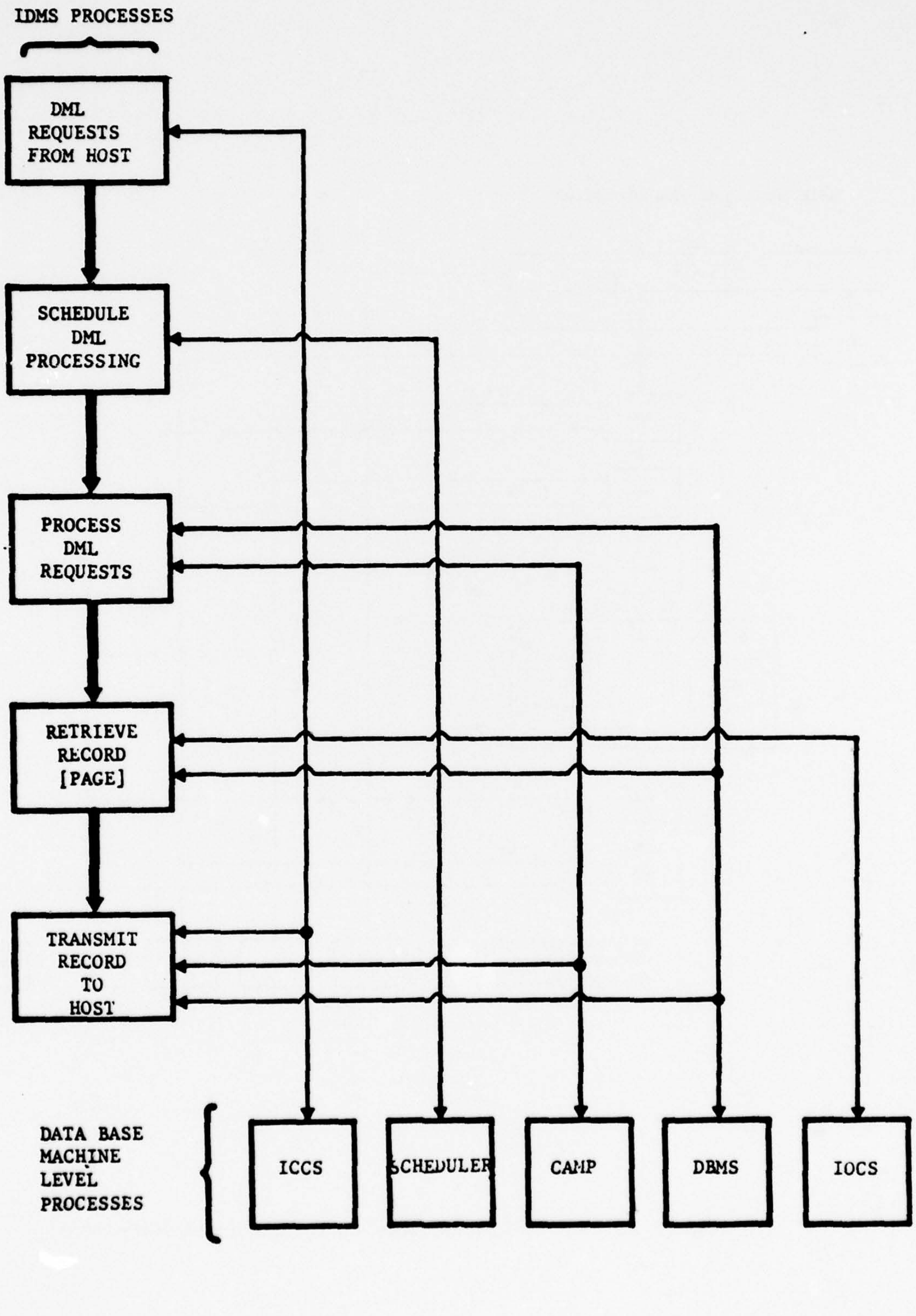


Figure A3.3-1 Relation of Data Base Machine Processes to IDMS Processes

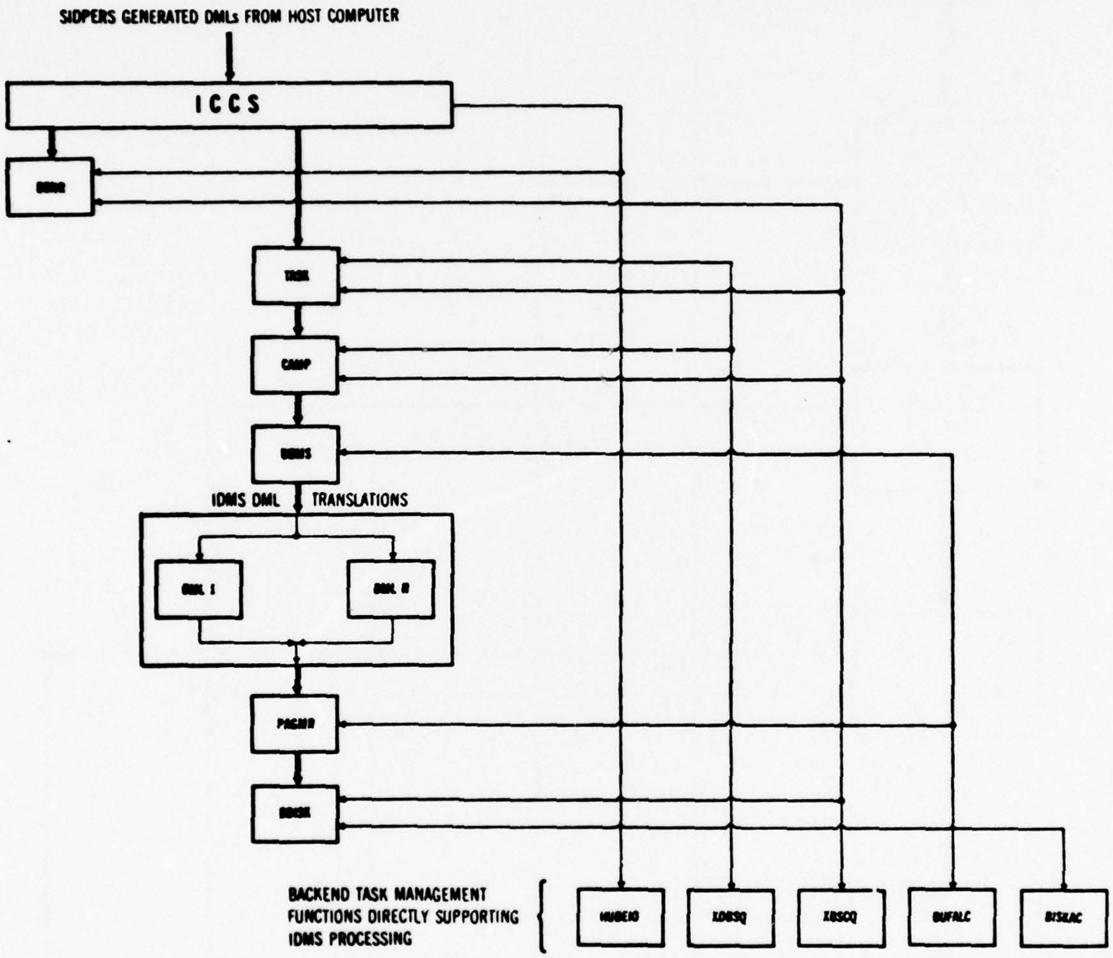


Figure A3.3-2 IPSS Realization of Back-end/IDMS Model

require communication with the host machine. This entails not only a service to characterize the backend interfacing functions such as the message printing and line protocol, but also a service to schedule the intercomputer communications. In the resulting IPSS submodel, this scheduling function was performed by the service, MUB910. It scheduled, in a FIFO manner, all requests to use the host/backend communication interface, from either the host or backend direction. It was assumed for this model that only one message could cross the interface at a time. Once the service HUBE10 scheduled and allocated the interface, the requesting service gained control of the use and disposition of the interface.

The second step in DML processing is the scheduling of IDMS itself. As mentioned before, the PDP-11/70 is not assumed to be decicated to the support of IDMS, allowing the concurrent execution of other non-IDMS applications. Thus, each invocation of IDMS, via entering DML command, must be scheduled and the appropriate IDMS software allocated. In the structure of IDMS its twin in the host application signals CAMP (Central Access Monitor Program) that a DML requires IDMS, and CAMP then schedules the invocation of IDMS itself. Only one DML can use the data base functions at a time to prevent concurrent updatings. The IPSS submodel for IDMS contained IPSS services corresponding to CAMP and the actual data base manipulation routines (DBMS). The signaling to CAMP that a DML request has arrived and needs DBMS is accomplished via the IPSS service XDBSQ. The actual allocation and deallocation occurs in CAMP.

Once the DBMS module is involved, the DML is processed in the third step via IDMS internal routing. Each of these routines has an analogue IPSS service that accomposes the high level DML into a sequence of primitive DBMS commands. IDMS individually changes these primitives to requests to its virtual data store for particular record(s) which leads to step 4.

The actual record retrieval in IDMS is a two step process. First, the record location is determined with respect to an IDMS page. The currently core resident pages are examined to see if one of them

contains the desired record. If none do, then a new page must be accessed. This page management function is accomplished via the IPSS service PAGMR (page manager). If a page must be accessed, then buffer space must be allocated (BUFACE) and the I/O request must be scheduled (XDSCO). When the I/O operations can be performed, the actual hardware devices for the PDP-11/70 backend computer are referenced via the IPSS service DISKB. Furthermore, DISKB simulates the allocation/deallocation of resources and the actual I/O operations of each and data transfer. DISKB also takes into account the dedicated data lists between main memory and auxiliary storage.

In order to characterize the backend operating system's functions as they normally operate, i.e., asynchronously, the IPSS IDMS and PDP-11/70 submodel also had to execute asynchronously. The host/backend interface (through HUBE1O) is also asynchronous. To accomplish this behavior, communication with operating system functions was characterized via the posting of common IPSS Event signals (semaphores). The IPSS built-in statements of WAIT EVENT cause the XBBSQ, HUBE1O, XBXCQ, XDSKQ services to remain in a wait state until needed. Conversely, once awakened by the IDMS services, these schedulers utilize the IPSS built-in statements FIND IN QUEUE, WAIT IN QUEUE and REMOVE FROM QUEUE to maintain queues of suspended IDMS DML requests.

A3.4 IDMS Models

The purpose of the model of IDMS was to characterize IDMS DML processing in order to determine:

1. the IDMS contribution to transaction response time,
2. the effect of widely varying workload definitions,
3. the effect of processing DMLs concurrently (through multiple terminal loadings), and
4. the effect of data structure on processing efficiencies.

Two models of IDMS were prepared, one employing the current IPSS facilities and the other incorporating advanced IPSS features for data

structure characterization and data base access. These two models are similar in structure, they contain the same services and the calling sequences remains unchanged. However, the IPSS/BASIC model does not make use of the DML parameters passed from the SIDPERS transaction services. Virtual page identification is characterized through probability distribution functions. Schema definition and data access currency are not represented. It was found that these important features of data base systems are very difficult to model without the special simulation facilities provided by IPSS/DBS.

The second model of IDMS represents these features through the advanced data structure and data access facilities of IPSS/DBS. In this model, the DML parameters passed by the application services are used to traverse the data base. The data base structure is characterized through the SCHEMA facility and instances of the data base are created through special purpose IPSS/DBS built-in functions. This results in a much more accurate determination of data base processing activities in general and virtual page referencing sequences in particular.
Secondary Storage functions were initially characterized by waiting an average elapsed time of 36 ms (the average time for PDP-11/70 head positioning and rotational delay). The detailed model of the PDP-11/70 was used to calculate these times.

Model Structure - Model I

The IPSS model of IDMS consists of services which interact in much the same way that the IDMS itself executes. The overall structure of the model is shown in Figure A3.4-1. The DML commands from the application services invoke the IDMS Interface service which in turn invokes the CAMP (Central Access Monitor Program) and DBMS services. The DBMS service invokes one of the 7 DML services which represent data access and retrieval. The interact with the Page Manager service and a single Schema Definition service in order to generate secondary storage references for the underlying file management system. The following is a description of the function of each of these services.

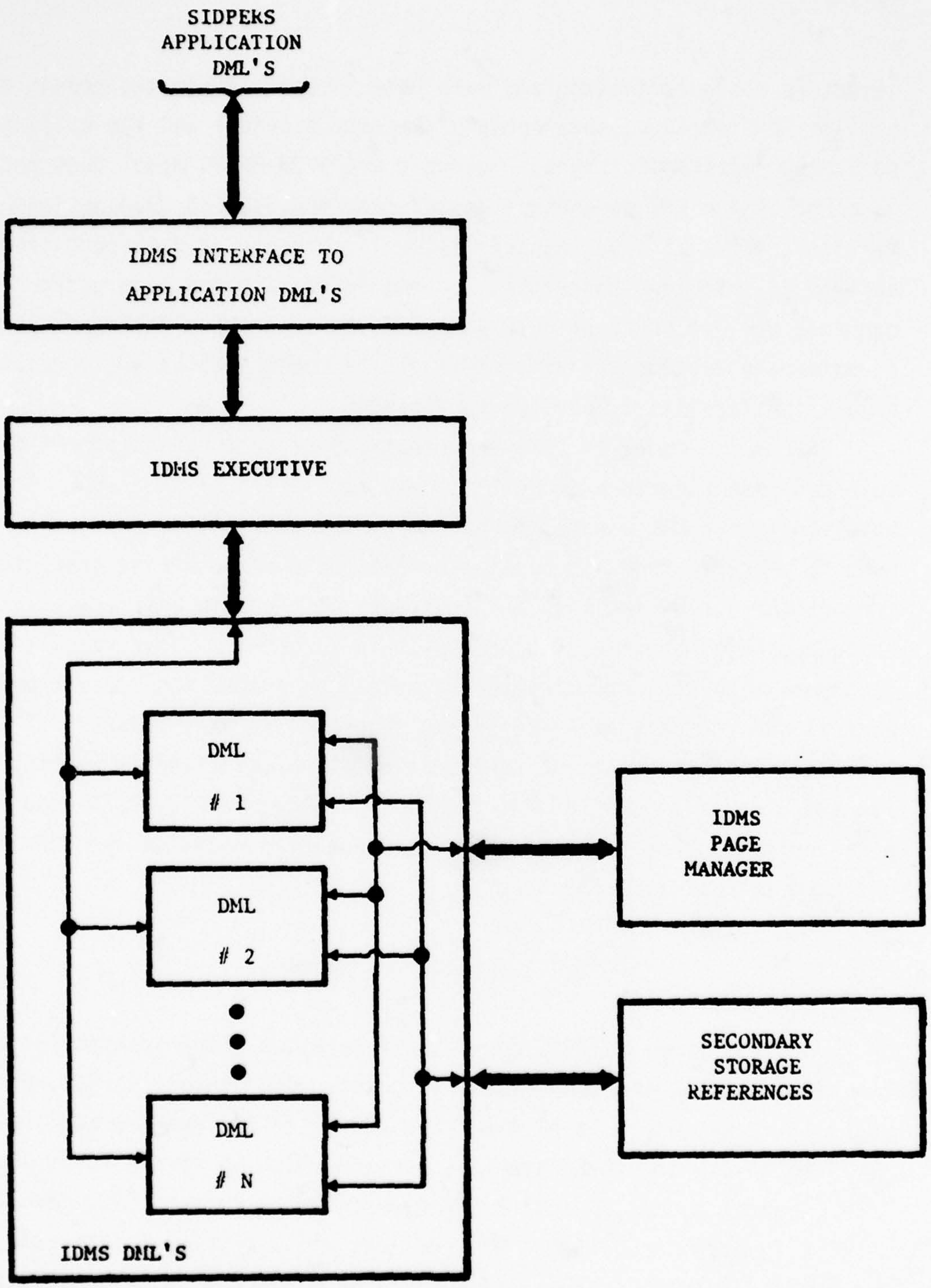


Figure A3.4-1 Schematic of the Modeled IDMS

IDMS Interface

This service represents IDMS processing activities that occur for every DML invocation. It is invoked by one of the application programs and represents the following IDMS functions:

1. validity checking and segmentation,
2. error status checking,
3. performing any 'before' procedures,
4. testing error status from the 'before' procedures,
5. testing the chanel indicator
6. testing error status on return from performing the data base management system function,
7. executing any "on error during" procedures,
8. executing any 'after error' procedures, and
9. performing segmentation.

However, no information was available on the frequency or nature of these procedures within the test VIABLE/SIDPERS system. Thus, these functions were merely identified and cocumented within this service and can thus be easily expanded once more data becomes available. This service invokes the CAMP Service and waits for its completion.

Central Access Monitor Program (CAMP) Service

This service represents the IDMS program of the same name. Its function is simply to thread requests as the DBMS service can accommodate them thus holding the invoking services in the wait state. Because of the single thread of processing represented by a single user terminal, the queueing of requests at this level does not occur. However, queueing is an important performance factor for models with more than one terminal and/or concurrently executing application programs.

Data Base Management System (DBMS) Service

This service represents those functions at the very heart of IDMS, namely the materialization of an application record. This invokes the schema and subschema definitions, data base currency, the content of

buffers, and (usually) requests to the operating system for one or more data base pages. Because of the magnitude and complexity of these processes, they are modeled through several IPSS services. The DBMS service establishes the environment (e.g., page size, buffer size), collects cache paping statistics, invokes one of the IPSS DML services and waits for its completion.

DML Services

There are 7 IDMS DMLs represented in the IPSS model, each through its own service definition. These DMLs are the IDMS functions identified in Table A2.3-4. Each of these services performs the logic of the DML by:

1. determining the virtual page number from the record tape and set parameters as well as currency,
2. determining if the requested page is already present in the IDMS page cache, and
3. initiating a service to retrieve the record if not present.

Model Structure - Model II

The IPSS/DBS model of IDMS is similar in structure to the IPSS/BASIC model reported in the previous section. As shown in Figure A3.4-1, the application DMLs invoke the CAMP service which in turn invokes the DBMS service. In this model, however, the DBMS service has the additional functions of creating an instance of the schema. This is accomplished by the CREATE SCHEMA built-in facility which copies the Data Base Structure Component schema definition into an internal work area. This service then creates a set of arrays (called routes) which represent schema access paths. These routes are the basic mechanism for creating instances of sets, maintaining currency, and simulating the traversal of the data base.

A route consists of a series of set identifiers which includes the identification of the owner and member record types. A single

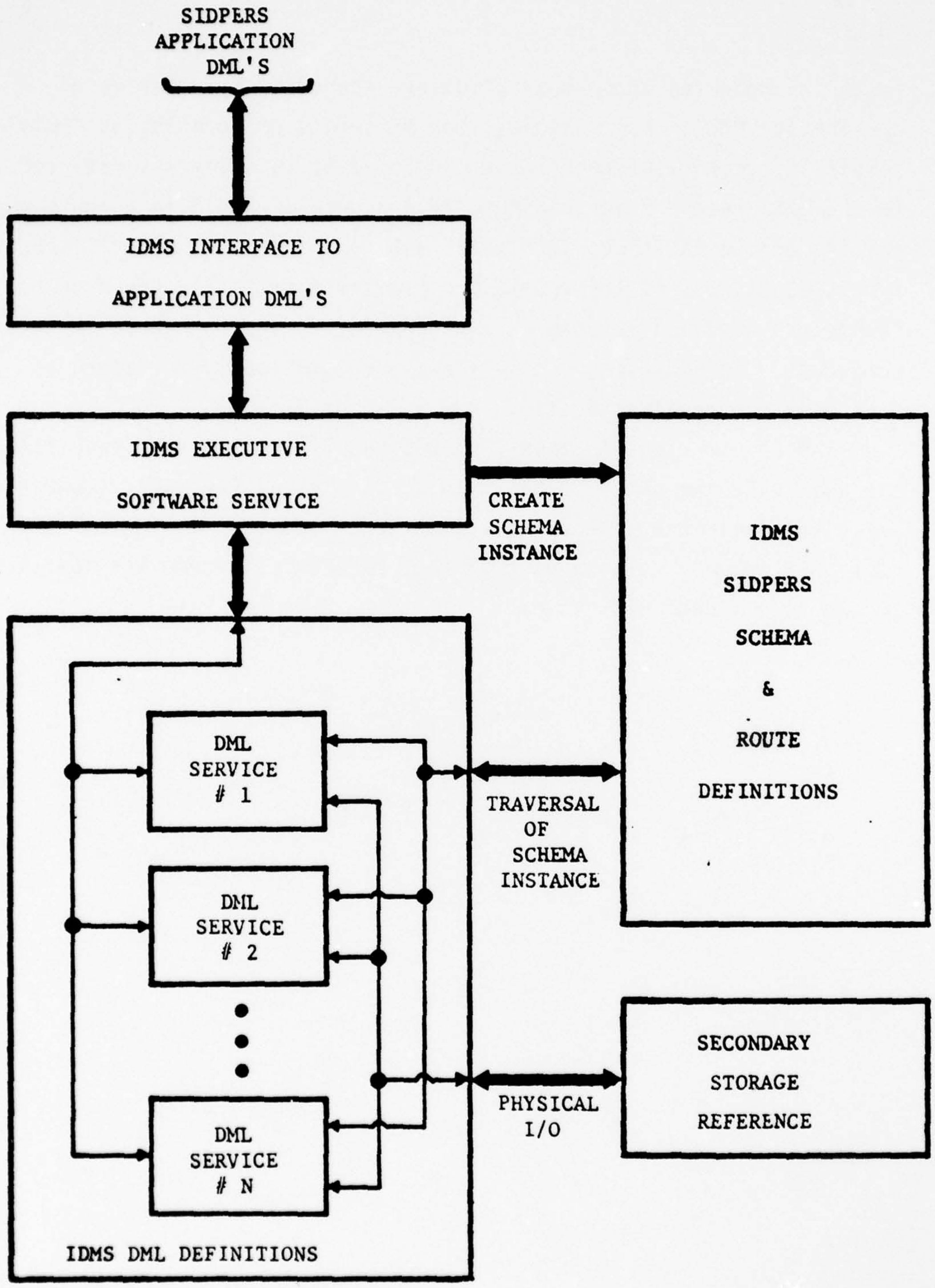


Figure A3.4-2 Schematic of IPSS/DBS Model of IDMS/SIDPERS Data Base

route incorporates those sets which are traversed by a series of application DMLs. For example, the DML series of locating an individual soldier's record via the CALC set followed by locating his data records in the UIST record type in Figure A2.3-2, would result in a route consisting of the CALC act, IDENT-UIST set, and the IDENT and UIST record types. Instances of these sets are created dynamically based on parameters supplied for the schema facility in the Data Base Structure component. In addition, statistics are automatically collected on data base traversal activities.

The DML services in this IPSS/DBS model simulate the traversal of the data base through such built-in facilities as FIND NEXT (member in set), and determine a record's virtual page address via the GET DATA BASE PAGE command. These virtual page references become the loading on the file management system.

APPENDIX B

SUMMARY OF MODEL STATISTICS

This appendix presents an overview and explanation of statistics generated by the execution of the IPSS models synthesized for the experimentation phase of the research. All of the defined models are not reproduced here, however, the final section of this Appendix contains the IPSS source listing for the SIDPERS/IDMS workload model.

B1. SIDPERS/IDMS WORKLOAD MODEL STATISTICS

The SIDPERS/IDMS workload model consists of a collection of IPSS service facilities whose invocation sequence is hierarchical. Figure B1-1 depicts the relationships among the services and indicates their generic function. As shown in the figure, transaction arrival at a work station is represented in the START service. Since a single terminal is modeled, the service queues all arriving transactions until the proceeding one has completed. Transaction parameters are assigned (TRANS) and the data fields are inputted and validated (VALID). Once the application programs is invoked (ADDSL, ARRIV, DEPTR, DTYC, GRCH, ING) which issues a DML by invoking the IDMS interface routine (modeled by INTF) and passing appropriate parameters. IDMS processing is sequenced by CAMP which then invokes DBMS. The function of the DBMS service is to establish the IDMS processing environment and to invoke a DML service (CONECT, DISCON, ERASE, AND, MODIFY, OBTAIN, STORE). These services identify an IDMS page and request its presence in core by invoking the PAGMR service. PAGMR manages the IDMS page buffer using a modified least-recently-used page replacement algorithm. In this model, data transfer from secondary storage is simulated by a clock advance of 36 ms. Table B1-1 relates the IPSS services identified in Figure B1-1 to the corresponding SIDPERS/IDMS activity.

Statistics relative to this modeled process are contained in the following pages. An elapsed time of two hours was simulated; all times

Table B1-1. Index to Modeled SIDPERS/IDMS Processes

Level	SIDPERS/IDMS Process	IPSS Endogenous Exogenous Service Name	Source Code Page
1	Arrival of transaction at a work station	START	3
2	Transaction input and verification		
	a. Transaction input by terminal operator	TRANS	4
	b. Transaction verification by system software	VALID	10
3	Application software processing		
	a. Add a soldier to the data base	ADDSL	31
	b. Process soldier's arrival at the installation	ARRIV	23
	c. Process soldier's departure from installation	DEPTR	27
	d. Process change in soldier's duty status	DTYC	18
	e. Process soldier's grade change	CRCH	13
	f. Process inquiry	INQ	21
4	Software processing performed by the IDMS interface routine	INTF	35
5	IDMS Central Access Monitor Processor (CAMP)	CAMP	37
6	IDMS DBMS processing	DBMS	39
7	IDMS DML processing		
	a. CONNECT DML	CONECT	50
	b. DISCONNECT DML	DISCON	51
	c. ERASE DML	ERASE	52
	d. FIND DML	FIND	45
	e. MODIFY DML	MODIFY	46
	f. OBTAIN DML	OBTAIN	42
	g. STORE DML	STORE	47
8	Page management, retrieval of pages from auxiliary storage	PAGMR	54

IPSS MODEL - SERVICE HIERARCHY	SERVICE FUNCTION
START	transaction arrival
TRANS VALID	transaction input and verification
ADDSL ARRIV DEPTR DTYC GRCH INQ	application processing
INTF	*IDMS interface routine*
CAMP	DML sequencer
DBMS	control
CONNECT DISCON ERASE FIND MODIFY OBTAIN STORE	DML's
PAGMR	page manager

Figure B1-1. SIDPERS/IDMS Workload Model Service Invocation Sequence

are reported in milliseconds. The first page of model output shows that the exogenous service START was busy 62.4% of the time and was invoked nineteen times. The mean number of transactions at the work station at any one time was 1.1 (the maximum number was 3) indicating that a single terminal is sufficient to handle the processing load anticipated for a small data base (203 soldiers).

The average elapsed time from transaction arrival at a work station to complete is 5.64 min (338,430 ms). An examination of the transit time statistics for the endogenous service facilities reveals that the majority of this time was spent in transaction input and validation (TRANS and VALID services). This is due to the relatively long data input time required and the terminal operator error rates reflected through these services.

A significant drop in the percent busy and the busy period mean length statistics is evidenced for the INTF service. This indicates that more terminals could easily be supported. An appropriate number could be determined by experimentation with the model parameters.

An examination of the DML statistics reveals a relatively large number of executions of these services and that the average processing time is low. For example, there were 83 invocations of the FIND DML service and the average processing time was 37.3 ms. Most of this time was consumed by the PAGMR service which all the DML's invoke. The PAGMR is required to bring pages into core, and as reflected in the cache page fault statistics, the miss ratio was high (94.7%). This indicates that the data base is not well organized with respect to the DML processing. As inspection of the DML's in the application program services, however, reveals that a majority of the processing is random via CALC sets, and hence there is very little chance of more optimally organizing the data base given this processing load.

The last page of statistics for this model displays the queueing of transactions for the DBMS and TRANS services. Since the DBMS service is single-threaded (the function of the CAMP service), no queueing to it occurred. However, if CAMP was multi-threaded and the DBMS was not,

no change would be required in the model to collect DBMS queueing statistics. The queueing statistics for the TRANS service shows that the queue was never very long (maximum of 2, mean length of .6) again indicating a larger data base or heavier transaction rate could easily be accommodated.

INFORMATION PROCESSING SYSTEM SIMULATOR

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = EXS

STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	START 1					
BUSY PERIOD STATISTICS						
PER CENT BUSY	62.4					
NUMBER	[illegible]					
MEAN LENGTH	499438.9					
STND DEV OF MEAN LENGTH	736315.3					
IDLE PERIOD STATISTICS						
PER CENT IDLE	37.6					
NUMBER	10					
MEAN LENGTH	270504.5					
STND DEV OF MEAN LENGTH	203298.0					
CONCURRENCY STATISTICS						
CURRENT LEVEL	0					
MAXIMUM LEVEL	3					
MEAN LEVEL	1.1					
STND DEV OF MEAN LEVEL	0.8					
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	19					
NUMBER OF ZERO TIME SEIZURES	0					
PER CENT ZERO TIME SEIZURES	0.0					
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	338430.0					
OVERALL STND DEV OF MEAN TIME	252403.6					
MEAN TIME*	338430.0					
STND DEV OF MEAN TIME*	252403.6					

*(EXCLUDING ZERO TIME SEIZURES)

INFORMATION PROCESSING SYSTEM SIMULATOR PAGE 3

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = ENS STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	ARRIV 1	CAMP 1	CONFCT 1	DEMS 1	DEPTR 1	DISCON 1
BUSY PERIOD STATISTICS						
PER CENT BUSY	0.2	0.4	0.1	0.4	0.0	0.0
NUMBER	18	574	60	574	2	27
MEAN LENGTH	951.8	44.2	108.6	44.2	667.5	97.3
STND DEV OF MEAN LENGTH	243.6	36.6	58.7	36.6	25.5	54.5
IDLE PERIOD STATISTICS						
PER CENT IDLE	99.8	99.6	99.9	99.6	100.0	100.0
NUMBER	19	575	61	575	3	28
MEAN LENGTH	378045.1	12477.6	117925.8	12477.6	2399554.0	257048.6
STND DEV OF MEAN LENGTH	670956.1	85898.4	318776.4	85898.4	3507626.0	474848.2
CONCURRENCY STATISTICS						
CURRENT LEVEL	0	0	0	0	0	0
MAXIMUM LEVEL	1	1	1	1	1	1
MEAN LEVEL	0.5	0.5	0.5	0.5	0.5	0.5
STND DEV OF MEAN LEVEL	0.5	0.5	0.5	0.5	0.6	0.5
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	18	574	60	574	2	27
NUMBER OF ZERO TIME SEIZURES	0	28	0	28	0	0
PER CENT ZERO TIME SEIZURES	0.0	4.9	0.0	4.9	0.0	0.0
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	951.8	44.2	108.6	44.2	667.5	97.3
OVERALL STND DEV OF MEAN TIME	243.6	36.6	58.7	36.6	25.5	54.5
MEAN TIME*	951.8	46.4	108.6	46.4	667.5	97.3
STND DEV OF MEAN TIME*	243.6	36.1	58.7	36.1	25.5	54.5
*(EXCLUDING ZERO TIME SEIZURES)						

INFORMATION PROCESSING SYSTEM SIMULATOR

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = ENS

STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	DTYC 1	ERASE 1	FIND 1	GRCH 1	INO 1	INTF 1
BUSY PERIOD STATISTICS						
PER CENT BUSY	0.0	0.0	0.0	0.1	0.0	0.4
NUMBER	4	6	83	3	4	574
MEAN LENGTH	413.1	36.0	37.3	1481.4	198.0	44.[illegible]
STND DEV OF MEAN LENGTH	54.0	0.0	8.8	20.8	20.[illegible]	36.[illegible]
IDLE PERIOD STATISTICS						
PER CENT IDLE	100.0	100.0	100.0	99.9	100.0	99.6
NUMBER	5	7	84	4	5	575
MEAN LENGTH	1439669.0	1028540.3	85677.1	1798888.0	1439841.0	12477.6
STND DEV OF MEAN LENGTH	1308099.0	895815.3	212961.4	1613929.0	1964420.0	55593.4
CONCURRENCY STATISTICS						
CURRENT LEVEL	0	0	0	0	0	0
MAXIMUM LEVEL	1	1	1	1	1	1
MEAN LEVEL	0.5	0.5	0.5	0.5	0.5	0.5
STND DEV OF MEAN LEVEL	0.5	0.5	0.5	0.5	0.5	0.[illegible]
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	4	6	83	3	4	574
NUMBER OF ZERO TIME SEIZURES	0	0	1	0	0	2[illegible]
PER CENT ZERO TIME SEIZURES	0.0	0.0	1.2	0.0	0.0	4.9
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	413.1	36.0	37.3	1481.4	198.0	44.[illegible]
OVERALL STND DEV OF MEAN TIME	54.0	0.0	8.8	20.8	20.8	36.6
MEAN TIME*	413.1	36.0	37.8	1481.4	198.0	46.4
STND DEV OF MEAN TIME*	54.0	0.0	7.8	20.8	20.8	36.1
*(EXCLUDING ZERO TIME SEIZURES)						

INFORMATION PROCESSING SYSTEM SIMULATOR

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = ENS STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	MODIFY 1	OBTAIN 1	PAGMR 1	STORE 1	TRANS 1	VALID 1
BUSY PERIOD STATISTICS						
PER CENT BUSY	0.0	0.2	0.3	0.0	62.4	62.1
NUMBER	62	319	699	17	19	34
MEAN LENGTH	14.7	37.6	35.0	0.0	236576.1	131456.0
STND DEV OF MEAN LENGTH	0.0	12.3	10.0	0.0	218316.4	138645.5
IDLE PERIOD STATISTICS						
PER CENT IDLE	100.0	99.8	99.7	100.0	37.6	37.9
NUMBER	63	320	700	18	20	35
MEAN LENGTH	114270.9	22462.4	10250.7	399999.6	135251.3	78011.3
STND DEV OF MEAN LENGTH	239647.5	114307.8	77980.0	692423.9	147061.9	162139.2
CONCURRENCY STATISTICS						
CURRENT LEVEL	0	0	0	0	0	0
MAXIMUM LEVEL	1	1	1	1	1	1
MEAN LEVEL	0.5	0.5	0.5	0.5	0.5	0.5
STND DEV OF MEAN LEVEL	0.5	0.5	0.5	0.5	0.5	0.5
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	62	319	699	17	19	34
NUMBER OF ZERO TIME SEIZURES	0	10	37	17	0	0
PER CENT ZERO TIME SEIZURES	0.0	3.1	5.3	100.0	0.0	0.0
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	14.7	37.6	35.0	0.0	236576.1	131458.0
OVERALL STND DEV OF MEAN TIME	0.0	12.3	10.0	0.0	218316.4	138645.5
MEAN TIME*	14.7	38.8	36.9	0.0	236576.1	131458.0
STND DEV OF MEAN TIME*	0.0	10.5	5.7	0.0	218316.4	138645.5
*(EXCLUDING ZERO TIME SEIZURES)						

```
           CACHE PAGE FAULT STATISTICS

CACHE SIZE (# PAGES):     5
PAGE SIZE (BYTES):      1024

PAGE WRITE REFERENCES:      17   CACHE MISSES:      0   MISS RATIO: 0.0
PAGE READ REFERENCES:      682   CACHE MISSES:    662   MISS RATIO: 0.9707

PAGE TOTAL REFERENCES:     699   CACHE MISSES:    662   MISS RATIO: 0.9471

PAGES REWRITEN BEFORE SPACE REUSED:       17
PAGES REPLACED WITHOUT REWRITING:        645

AVERAGE DISTANCE BETWEEN SUCCESSIVE PAGE REFERENCES:    37.1539
```

INFORMATION PROCESSING SYSTEM SIMULATOR

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART D - QUEUEING STATISTICS

FACILITY TYPE = ENS

STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	DBMS 1	TRANS 1				
BUSY PERIOD STATISTICS						
PER CENT BUSY	0.0	23.9				
NUMBER	574	18				
MEAN LENGTH	0.0	95422.4				
STND DEV OF MEAN LENGTH	0.0	144971.9				
IDLE PERIOD STATISTICS						
PER CENT IDLE	100.0	76.1				
NUMBER	575	19				
MEAN LENGTH	12521.6	288546.6				
STND DEV OF MEAN LENGTH	85898.6	195194.2				
QUEUE LENGTH STATISTICS						
CURRENT LENGTH	0	0				
MAXIMUM LENGTH	1	2				
MEAN LENGTH	0.5	0.6				
STND DEV OF MEAN LENGTH	0.5	0.6				
QUEUE ENTRY STATISTICS						
NUMBER OF COMPLETED ENTRIES	574	19				
NUMBER OF ZERO TIME ENTRIES	574	9				
PER CENT ZERO TIME ENTRIES	100.0	47.4				
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	0.0	101853.7				
OVERALL STND DEV OF MEAN TIME	0.0	142386.9				
MEAN TIME*	0.0	193522.1				
STND DEV OF MEAN TIME*	0.0	144350.6				
*(EXCLUDING ZERO TIME SEIZURES)						

B2. HOST/SIDPERS SUBMODEL STATISTICS

Previous USACSC studies did not identify a particular machine to be used for the host environment, so a generalized host was modeled. To do this, the software functions of the host were emphasized more than the hardware functions. Thus, only a minimum hardware configuration was described and more attention was paid towards determining and modeling the host operating system requirements of the SIDPERS application.

The host operating system was assumed to execute in a reactive manner, responding to the needs of the SIDPERS application. The particular steps in the modeled SIDPERS application were discussed in the main body of this report and in Appendix A. The needs of this application were, in general, task management and resource allocation. A crucial host/SIDPERS interface was the synchronization of the execution of SIDPERS services and operating system services. The asynchronous nature of their execution compounded the problem. To model this situation, IPSS services representing task schedulers communicated with existing SIDPERS IPSS services via the posting of and waiting for global events. These events corresponded to a request for scheduling and the completion of that scheduling process.

Four schedulers were identified for the modeling of the host/SIDPERS interface. These controlled access to the host's disk subsystem, the use of the teleprocessing(TP) line to the user, the use of the teleprocessing line to the back-end and the resumption of execution after a back-end request was completed. Each of these were characterized by a separate IPSS service and their use was controlled through a queue, exemplified by an IPSS chain facility. When a SIDPERS transaction required scheduling, it was placed in one of these chains. The four chains were HDSKQ (disk subsystem), TCAMQ (user TP communication), HOBEQ (back-end TP communication) and RSPONS (back-end request completion).

The use of these chains was measured in IPSS via the queueing statistics shown. Because of the particular SIDPERS workload, no SIDPERS applications overlapped, and, therefore, there was no queue in transit time associated with any scheduler. This condition also limited the possibility of having more than one SIDPERS transaction enqueued simultaneously. If such a situation occurred in the future, the concurrency statistics would demonstrate the degree of delay.

The queue entry statistics indicate the number of times that the current SIDPERS workload requested each of the scheduling tasks. The SIDPERS load for the run depicted is shown in the utilization statistics for the IPSS services that characterize SIDPERS processing. The total number of SIDPERS transaction for this run was 9, i.e., the number of requests to VALID. The number of requests to HDSKQ corresponds to the number of validity and/or compatability errors that occurred during the current run, since each error required a sisk access. Thus, on the average, each incoming transaction had one error.

Statistics measuring the communication with the user and the back-end indicate the complexity of the SIDPERS application transactions. The number of user interactions (TCAMQ) reflects the amount of input required for the current set of SIDPERS transactions. Thus, for 9 transactions, a mean of 5.1 interactions were required per transaction. This included input of data and output of error messages and records. The number of entries in the RSPONS queue indicates that the current input stream of SIDPERS transactions generated 145 DML request to the back-end. This implies a mean of 16.1 DML requests per SIDPERS transaction types and does not take into account the DML requirements of particular transaction types. And finally, the number of entries in the back-end interface chain, HOBEQ, is twice the number of the actual DML requests since the TP interface must be acquired for both sending and receiving DML requests.

INFORMATION PROCESSING SYSTEM SIMULATOR PAGE 116

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART D - QUEUEING STATISTICS

FACILITY TYPE = CHN

STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	HDSKO 1	HOBEQ 1	RSPONS 1	TCAMQ 1		
BUSY PERIOD STATISTICS						
PER CENT BUSY	0.0	0.0	0.5	0.0		
NUMBER	10	290	145	46		
MEAN LENGTH	0.0	0.0	100.0	0.0		
STND DEV OF MEAN LENGTH	0.0	0.0	0.0	0.0		
IDLE PERIOD STATISTICS						
PER CENT IDLE	100.0	100.0	99.5	100.0		
NUMBER	11	291	146	47		
MEAN LENGTH	254545.1	9621.9	19078.7	59574.2		
STND DEV OF MEAN LENGTH	571445.4	69778.4	97742.3	140279.8		
QUEUE LENGTH STATISTICS						
CURRENT LENGTH	0	0	0	0		
MAXIMUM LENGTH	1	1	1	1		
MEAN LENGTH	0.5	0.5	0.5	0.5		
STND DEV OF MEAN LENGTH	0.5	0.5	0.5	0.5		
QUEUE ENTRY STATISTICS						
NUMBER OF COMPLETED ENTRIES	10	290	145	46		
NUMBER OF ZERO TIME ENTRIES	10	290	0	46		
PER CENT ZERO TIME ENTRIES	100.0	100.0	0.0	100.0		
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	0.0	0.0	100.0	0.0		
OVERALL STND DEV OF MEAN TIME	0.0	0.0	0.0	0.0		
MEAN TIME*	0.0	0.0	100.0	0.0		
STND DEV OF MEAN TIME*	0.0	0.0	0.0	0.0		

*(EXCLUDING ZERO TIME SEIZURES)

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = ENS STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	ARRIV 1	BACK 1	DEPTR 1	DTYC 1	INQ 1	INTF 1
BUSY PERIOD STATISTICS						
PER CENT BUSY	0.3	0.5	0.1	0.1	0.0	0.5
NUMBER	4	145	1	2	2	145
MEAN LENGTH	2204.4	100.0	2104.2	1202.4	601.2	100.2
STND DEV OF MEAN LENGTH	231.4	0.0	0.0	0.0	0.0	0.3
IDLE PERIOD STATISTICS						
PER CENT IDLE	99.7	99.5	99.9	99.9	100.0	99.5
NUMBER	5	146	2	3	3	146
MEAN LENGTH	558236.2	19076.7	1390947.0	932531.0	932432.0	19078.5
STND DEV OF MEAN LENGTH	814536.6	97742.3	241210.1	342462.8	1159113.0	97742.4
CONCURRENCY STATISTICS						
CURRENT LEVEL	0	0	0	0	0	0
MAXIMUM LEVEL	1	1	1	1	1	1
MEAN LEVEL	0.5	0.5	0.5	0.5	0.5	0.5
STND DEV OF MEAN LEVEL	0.5	0.5	0.7	0.6	0.6	0.5
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	4	145	1	2	2	145
NUMBER OF ZERO TIME SEIZURES	0	0	0	0	0	0
PER CENT ZERO TIME SEIZURES	0.0	0.0	0.0	0.0	0.0	0.0
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	2204.4	100.0	2104.2	1202.4	601.2	100.2
OVERALL STND DEV OF MEAN TIME	231.4	0.0	0.0	0.0	0.0	0.3
MEAN TIME*	2204.4	100.0	0.0	1202.4	601.2	100.2
STND DEV OF MEAN TIME*	231.4	0.0	0.0	0.0	0.0	0.3
*(EXCLUDING ZERO TIME SEIZURES)						

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = ENS STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	TRANS 1	VALID 1				
BUSY PERIOD STATISTICS						
PER CENT BUSY	23.4	22.9				
NUMBER	7	10				
MEAN LENGTH	93523.5	64008.0				
STND DEV OF MEAN LENGTH	119212.9	106357.9				
IDLE PERIOD STATISTICS						
PER CENT IDLE	76.6	77.1				
NUMBER	7	10				
MEAN LENGTH	306476.1	215991.7				
STND DEV OF MEAN LENGTH	229362.3	238902.2				
CONCURRENCY STATISTICS						
CURRENT LEVEL	1	1				
MAXIMUM LEVEL	1	1				
MEAN LEVEL	0.5	0.5				
STND DEV OF MEAN LEVEL	0.5	0.5				
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	6	9				
NUMBER OF ZERO TIME SEIZURES	0	0				
PER CENT ZERO TIME SEIZURES	0.0	0.0				
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	60278.0	71089.4				
OVERALL STND DEV OF MEAN TIME	88146.7	110280.8				
MEAN TIME*	60278.0	71089.4				
STND DEV OF MEAN TIME*	88146.7	110280.8				
*(EXCLUDING ZERO TIME SEIZURES)						

B3. IDMS/BACK-END SUBMODEL STATISTICS

The emphasis of this model was on interfacing the IDMS functions with the PDP-11/70 backend operations. Because of the wealth of documentation available on the hardware aspects of the USACSC PDP-11/70 (and the lack of software documentation), the impact of IDMS on hardware utilization was examined. The complete configuration of the USACSC PDP-11/70 installation at the Georgia Institute of Technology was modeled and is shown in Figure B3-1.

The PDP-11/70 operating system employs a dedicated data path between central memory and high speed disks (RWPO4AA in Figure B3-1). The IDMS data base resides on these disks, allowing for rapid retrieval of IDMS pages. DML requests entering the backend from the host were eventually mapped to specific data accesses to the RWPO4AA. Thus, to measure the direct effect of the IDMS loading on the PDP-11/70 backend, the IPSS statistics on the behavior of this disk subsystem (labeled RPO4AA in the corresponding IPSS model) are examined and are found immediately following this discussion.

Utilization statistics on the RPO4AA provide the best insight into its performance relative to the IDMS loading. Busy/idle statistics indicate the percentage of time that this facility was utilized. As can be seen, the RPO4AA was used only 23.1% of the time, remaining idle for 76.9%. Such an imbalance can be attributed to the characteristics of the device exceeding the needs of the application or the actual placement of data on the disks.

Concurrency statistics measure the degree of multiprocessing that occurred on the RPO4AA. In this case, in order to maintain the integrity of the IDMS data base, only one access at a time was permitted. This is evidenced by the value of the maximum level of concurrency being 1. The value of the current levels equalling 1 implies that at the end of the current simulation run, the RPO4AA device was in use.

The last two sets of utilization statistics characterize the behavior of individual accesses to the RPO4AA. Each access involves a seek and a data transfer operation. The RPO4AA was acquired (seized) throughout both operations. The total number of references(19) and the number

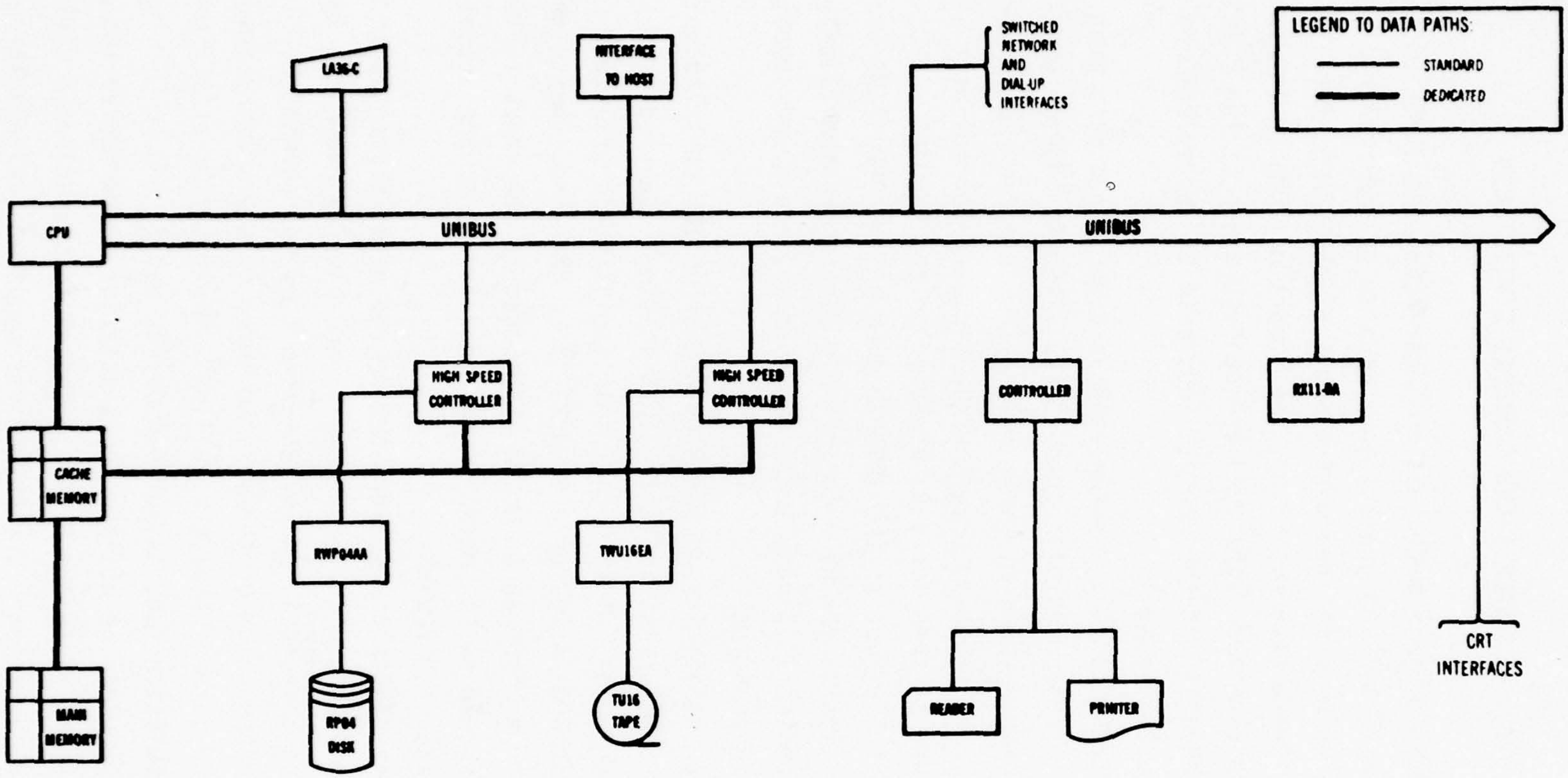


Figure B3-1 USACSC PDP 11/70 Configuration

of zero time references (0) indicate the relative closeness of successive page references to the IDMS data base. A low percentage, as is the case here, indicate that the data may be well spread out on the disk or that the individual accesses are randomly distributed. On the other hand, a high percentage of zero time seizures may indicate the compactness of the data or a deliberate ordering of accesses (e.g., sequential of within an IDMS set). The mean total time for an individual accesses to the RP04AA, 23.2 milliseconds, includes the seek and data transfer operations. An additional mean is calculated removing the bias of the zero time seizures so that an average time per actual access can be determined. For each mean, the variability of that mean, i.e., the standard deviation, is also calculated to give some insight into the relative accuracy of the above means.

The impact of the IDMS functions can also be shown in other IPSS statistics. The IDMS DML requests are mapped into I/O requests for which IPSS has built-in primitives such as Seek and Data Transfer. Statistics are automatically calculated by IPSS on invocations of each of these I/O operations by each I/O operation (Part B1), by access mechanism (Part B2) and by data set (Part B3). The statistics by the access mechanism, RP04AA, aid in analyzing the impact of IDMS on the backend. The sum of the mean routine times for the two I/O operations that referenced RP04AA equals 23.2 milliseconds, the same utilization time for RP04AA in the previous discussion. This allows us to break down the aggregate access time to RP04AA into its component activities. It is evident that the Seek operation comprises the greatest amount of time in each DML request processing. Thus, in order to reduce this time factor affecting the Seek operation, such as placement of data files, compactness or device type, should be examined.

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART C - UTILIZATION STATISTICS

FACILITY TYPE = AA STATISTICS DESCRIPTION	FACILITY 1 NAME INDEX	FACILITY 2 NAME INDEX	FACILITY 3 NAME INDEX	FACILITY 4 NAME INDEX	FACILITY 5 NAME INDEX	FACILITY 6 NAME INDEX
	RF04AA 1					
BUSY PERIOD STATISTICS						
PER CENT BUSY	23.1					
NUMBER	20					
MEAN LENGTH	23.4					
STND DEV OF MEAN LENGTH	12.4					
IDLE PERIOD STATISTICS						
PER CENT IDLE	76.9					
NUMBER	20					
MEAN LENGTH	78.0					
STND DEV OF MEAN LENGTH	13.4					
CONCURRENCY STATISTICS						
CURRENT LEVEL	1					
MAXIMUM LEVEL	1					
MEAN LEVEL	0.5					
STND DEV OF MEAN LEVEL	0.5					
SEIZURE STATISTICS						
NUMBER OF COMPLETED SEIZURES	19					
NUMBER OF ZERO TIME SEIZURES	0					
PER CENT ZERO TIME SEIZURES	0.0					
TRANSACTION TRANSIT TIME STATISTICS						
OVERALL MEAN TIME	23.2					
OVERALL STND DEV OF MEAN TIME	12.7					
MEAN TIME*	23.2					
STND DEV OF MEAN TIME*	12.7					
*(EXCLUDING ZERO TIME SEIZURES)						

INFORMATION PROCESSING SYSTEM SIMULATOR　　PAGE 2

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART AI - I/O STATISTICS BY I/O ROUTINE

STATISTICS DESCRIPTION	TOTAL	ZERO	%ZERO	MEAN	STD DEV	*MEAN	*STD DEV
SEEK STATISTICS							
NUMBER COMPLETED	19	5	26.3				
TIME IN EXECUTION				20.6	12.7	28.0	0.0
NUMBER OF CYLINDERS CROSSED				2.3	2.2	3.1	1.9
DATA TRANSFER STATISTICS							
DEVICE TYPE - DASD							
NUMBER OF READS	19						
NUMBER OF WRITES	0						
TIME IN EXECUTION				2.0	0.0		
NUMBER OF CHARACTERS TRANSMITTED				1024.0	0.0		

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART F2 - I/O STATISTICS BY ACCESS MECHANISM

FACILITY IDENTIFICATION		I/O ROUTINE	NUMBER OF I/O CALLS			ROUTINE TIME			
NAME	INDEX	NAME	TOTAL	ZERO	%ZERO	MEAN	STD DEV	*MEAN	*STD DEV
RP04AA	1	SEEK	19	5	26.3	20.6	12.7	28.0	0.0
		DATA TRANS	19	0	0.0	2.6	0.0	2.6	0.0

INFORMATION PROCESSING SYSTEM SIMULATOR

MODEL SIMULATION PHASE

POST SIMULATION STATISTICS

PART 13 - I/O STATISTICS BY DATA SET . FILE

FACILITY IDENTIFICATION			I/O ROUTINE	NUMBER OF I/O CALLS			ROUTINE TIME			
NAME	INDEX	FILE	NAME	TOTAL	ZERO	%ZERO	MEAN	STD DEV	*MEAN	*STD DEV
[illegible]	1	1	[illegible]	19	0	0.0	2.0	0.0	2.0	0.0

B4. IPSS FUNCTIONAL MODEL OF THE SIDPERS/IDMS

The purpose of this section is to illustrate the use of the IPSS to describe a complex information processing system at the functional level. This model does not contain characterizations for the computer hardware, executive software, data base management system software or data base. At this level they were treated as black boxes, however appropriate interfaces were defined so that respective characterizations could be included in later models.

The focus of this model was to model the processes (represented as IPSS endogenous services) encountered in updating the SIDPERS data base. Figure B1-1 identified the hierachical relationship assumed for the model; Table B1-1 related the IPSS service identified in Figure B1-1 to a corresponding SIDPERS/IDMS data base update procedure. The IDMS data base management system backend was also represented in the model.

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                            PAGE    1
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                         INPUT CARD IMAGE                                       INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

   DEFINE SYSTEM RESOURCES: NAME=CF, COMPILE=YES, DISPOSITION=KEEP;  00000010       1
   PROCEDURE: NAME=COMENT, TYPE=SUBROUTINE;                          00000020       2               1
C  MODEL OF- USACSC SIDPERS IDMS TEST DATA BASE     -PERS3-          00000030       3               2
C  FOR- U.S. ARMY COMPUTER SYSTEM COMMAND                            00000040       4               3
C  APPLICATION- SIX TYPES OF TRANSACTION PROCESSING -  GRADE CHANGE, 00000050       5               4
C     DUTY STATUS CHANGE, INQUIRY, ARRIVAL, DEPARTURE, ADD SOLDIER   00000060       6               5
C  WRITTEN BY- JOSEPH D. BROWNSMITH                                  00000070       7               6
C  DATE-  JUNE-AUG 1977                                              00000080       8               7
C  NOTE- THIS MODEL IS WRITTEN IN IPSS (INFORMATION PROCESSING SYSTEM00000090       9               8
C        SYSTEM SIMULATOR) AND REPRESENTS THE USACSC'S SIDPERS IDMS  00000100      10               9
C        TEST DATA BASE SYSTEM RUNNING ON THE CSC IBM370/165 COMPUTER 00000110     11              10
C        (FRONT-END) AND THE PDP11/70 (BACK-END).                    00000120      12              11
C                                                                    00000130      13              12
C  THIS SIMULATION INCORPORATES THE FOLLOWING EXPERIMENT(S) -        00000140      14              13
C    I                                                               00000150      15              14
C                                                                    00000160      16              15
C  EXPERIMENT #1                                                     00000170      17              16
C     STANDARD SIDPERS TRANSACTION AND IDMS PARAMETERS               00000180      18              17
C  EXPERIMENT #2                                                     00000190      19              18
C     PVERR = PCERR = 0 (PROB VALIDITY & COMPATIBILITY ERRORS)       00000200      20              19
C        (SEE LINES 2540-2590)                                       00000210      21              20
C  EXPERIMENT #3                                                     00000220      22              21
C     AVEDIS = 10 (DISTANCE BETWEEN SUCCESSIVE PAGE REFERENCES)      00000230      23              22
C        (SEE LINE 9670)                                             00000240      24              23

........10........20........30........40........50........60........70........80
```

```
                                INFORMATION PROCESSING SYSTEM SIMULATOR                                   PAGE    2
                                      STANDARD INPUT STREAM LISTING
                                           DATE ... 09/10/77
                                 INPUT CARD IMAGE                                                INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80      SEQ-NO  LEV SEQ-NO   REF-NO

C                                                                       00000250          25                  24
C     THE MODEL CONSISTS OF THE FOLLOWING COMPONENTS -                  00000260          26                  25
C   REF   SERVICE/PROCEDURE    APPROX    INVOKES/                       00000270          27                  26
C    #          NAME            PAGE      CALLS         DESCRIPTION     00000280          28                  27
C   ---   --------------------  ----     ------   ----------------------00000290          29                  28
C                                                                       00000300          30                  29
C    1   SYSTEM RESOURCES       (01)        -     COMPONENT DEFINITION  00000310          31                  30
C    2     COMENT (PROC)        (01)        -       COMMENTS            00000320          32                  31
C    3     START (EXS)          (03)        4       SCHEDULE TRANS (EV)100000330          33                  32
C    4     TRANS (ENS)          (04)      5-11      TRANS SELECTION     00000340          34                  33
C    5     VALID (ENS)          (10)        -       TERMINAL INTERACTION00000350          35                  34
C    6     GRCH (ENS)           (13)       12       GRADE CHANGE        00000360          36                  35
C    7     DTYC (ENS)           (18)       12       DUTY STATUS         00000370          37                  36
C    8     INQ (ENS)            (21)       12       INQUIRY             00000380          38                  37
C    9     ARRIV (ENS)          (23)       12       ARRIVAL             00000390          39                  38
C   10     DEPTR (ENS)          (27)       12       DEPARTURE           00000400          40                  39
C   11     ADDSL (ENS)          (31)       12       ADD SOLDIER         00000410          41                  40
C   12     INTF (ENS)           (35)       13       HOST INTERFACE      00000420          42                  41
C   13     CAMP (ENS)           (37)       14       ACCESS MONITOR      00000430          43                  42
C   14     DBMS (ENS)           (39)     15-21      DBMS                00000440          44                  43
C   15     OBTAN (ENS)          (42)     22-23      OBTAIN DML          00000450          45                  44
C   16     FIND (ENS)           (45)     22-23      FIND DML            00000460          46                  45
C   17     MODIFY (ENS)         (46)     22-23      MODIFY DML          00000470          47                  46
C   18     STORE (ENS)          (47)     22-23      STORE DML           00000480          48                  47

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                          PAGE    3
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                     INPUT CARD IMAGE                                       INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

C  19     CONECT (ENS)        (50)     22-23     CONNECT DML         00000490      49                  48
C  20     DISCON (ENS)        (51)     22-23     DISCONNECT DML      00000500      50                  49
C  21     ERASE (ENS)         (52)     22-23     ERASE DML           00000510      51                  50
C  22     PAGMR (ENS)         (54)       -       PAGE MANAGER (CACHE)00000520      52                  51
C  23     GENRP (PROC)        (57)       -       PAGE # GENERATOR    00000530      53                  52
C  24     BUFST (EXS)         (60)       -       PRINT PAGE STAT(EV2)00000540      54                  53
C  25     SNAPST (EXS)        (61)       -       PRINT IPSS STAT(EV3)00000550      55                  54
C  26   EXO EVENT STREAM      (61)       -     COMPONENT DEFINITION  00000560      56                  55
C  27     EV1 (EXO EVENT)     (62)       3       INVOKE PROCESSING   00000570      57                  56
C  28     EV2 (EXO EVENT)     (62)      24       DISPLAY PAGE STATS  00000580      58                  57
C  29     EV3 (EXO EVENT)     (62)      25       DISPLAY IPSS STATS  00000590      59                  58
C  30   MODEL                 (65)       -     COMPONENT DEFINITION  00000600      60                  59
C                                                                    00000610      61                  60
C                                                                    00000620      62                  61
C      RT       IDMS ID     #          SET     IDMS IDENTIFIER       00000630      63                  62
C     ----------------------          ---------------------------    00000640      64                  63
C     ADDI    ADD-IDENT    410      1. CALC    CALC                  00000650      65                  64
C     ADUN    ADD-UNIT     420      2. GRUI    GRADE-UIST            00000660      66                  65
C     ALOC    AALOC        455      3. IDUI    IDENT-UIST            00000670      67                  66
C     GRADE   GRADE        350      4. INUM    INST-UNIT-MASTER      00000680      68                  67
C     IDEN    IDENT        320      5. PMID    IDENT-MOSC (PMOS)     00000690      69                  68
C     INST    INST         300      6. PDUI    POSNO-UIST            00000700      70                  69
C     MOSC    MOSC         460      7. SMID    IDENT-MOSC (SMOS)     00000710      71                  70
C     POSN    POSNO        440      8. UMAW    UNIT-MASTER-AWOL      00000720      72                  71

........10........20........30........40........50........60........70........80
```

```
                          INPUT CARD IMAGE                                  INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

C     UIST    UIST        330      9. UMGR    UNIT-MASTER-GRADE     00000730      73                  72
C     UNMA    UNIT-MASTER 310     10. UMPOS   UNIT-MASTER-POSNO     00000740      74                  73
C                                 11. UMUI    UNIT-MASTER-UIST      00000750      75                  74
      RETURN                                                        00000760      76                  75
      END: PROCEDURE;                                               00000770      77                  76
                                                                    00000780      78
                                                                    00000790      79
      EXO SERVICE: ID=START, NAME=STARTS,                           00000800      80                  77
                    SAVE AREA SIZE = 10;                            00000810      81                  78
C                                                                   00000820      82                  79
C     THIS SERVICE REPRESENTS THE ARRIVAL OF TRANSACTIONS AT A SINGLE   00000830  83                  80
C     WORK STATION. IT QUEUES THE TRANSACTION UNTIL THE PREVIOUS    00000840      84                  81
C     TRANSACTION IS COMPLETED.                                     00000850      85                  82
      END: DECLARATIONS;                                            00000860      86                  83
      END: INITIALIZATION;                                          00000870      87                  84
      SEIZE: FACILITY=START;                                        00000880      88                  85
        QUEUE: FACILITY=TRANS;                                      00000890      89                  86
        WAIT FACILITY: FACILITY=TRANS, STATUS=0, CONDITION=EQ;      00000900      90                  87
        DEPART QUEUE:  FACILITY=TRANS;                              00000910      91                  88
        SET STATUS: FACILITY=TRANS, STATUS=1;                       00000920      92                  89
          INVOKE: SERVICE=TRANS$;                                   00000930      93                  90
          WAIT RETURN: SERVICE=TRANS;                               00000940      94                  91
        SET STATUS: FACILITY=TRANS, STATUS=0;                       00000950      95                  92
      RELEASE: FACILITY=START;                                      00000960      96                  93

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                         INPUT CARD IMAGE                                      INPUT   ALT INPUT  MEMBER
.........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

      END: EXO SERVICE;                                                00000970   97              94
                                                                       00000980   98
                                                                       00000990   99
      ENDO SERVICE: ID=TRANS, NAME=TRANS$,                             00001000  100              95
                   SAVE AREA SIZE = 10;                                00001010  101              96
C                                                                      00001020  102              97
C     THIS SERVICE REPRESENTS THE INTERACTION OF TERMINAL USER ENTERING 00001030 103              98
C     TRANSACTIONS INTO THE SIDPERS SYSTEM.  THIS TERMINAL CONTROL     00001040  104              99
C     PROCESSING MODULE INCLUDES TERMINAL INTERACTION WITH THE DATA BASE00001050 105             100
C     'VALID' RECORDS AND ALSO VALIDITY CHECKS.  THE CHARACTERIZATION  00001060  106             101
C     OF TRANSACTION PROCESSING REPRESENTED IN THIS SERVICE WAS        00001070  107             102
C     DERIVED FROM "DESCRIPTION OF THE VIABLE TRANSACTIONS FOR THE     00001080  108             103
C     DIMUI MODEL" BY HELMUT SCHAAFF, MARCH 1977.                      00001090  109             104
C                                                                      00001100  110             105
C     THE TYPES OF TRANSACTIONS REPRESENTED IN THE MODEL AND THEIR     00001110  111             106
C     PERCENT OCCURRENCE ARE-                                          00001120  112             107
C        1.  GRADE CHANGE         6.68%    4.  ARRIVAL     20.31%       00001130  113             108
C        -.  MOS CHANGE (NO DATA)          5.  DEPARTURE 24.81%        00001140  114             109
C        2.  DUTY STATUS CHANGE  42.55%    6.  ADD          0.77%.     00001150  115             110
C        3.  INQUIRY              4.88%                                00001160  116             111
C                                                                      00001170  117             112
      DATA SERVCE/'TRAN'/                                              00001180  118             113
C                                                                      00001190  119             114
        INTEGER RETC, SEED, SUB, TLOC(10)                              00001200  120             115
.........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                        INPUT CARD IMAGE                                      INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

         REAL RN, SPMLIN, SUNFM                                     00001210   121              116
         REAL POINTS (7,2)                                          00001220   122              117
         COMMON /WK/ TLOC                                           00001230   123              118
C        DISTRIBUTION TABLE FOR THE TRANSACTION TYPES               00001240   124              119
         DATA POINTS/1.0, 2.0,    3.0,    4.0,    5.0,    6.0,    6.0,  00001250   125              120
     I           0.0, 0.0668, 0.4923, 0.5411, 0.7442, 0.9923, 1.0/  00001260   126              121
C                                                                   00001270   127              122
C        TRANSACTION PARAMETERS                                     00001280   128              123
C        1 - PERCENT DIRECT MODE                                    00001290   129              124
C        2 - NUMBER OF DATA ITEMS PER TRANSACTION ENTRY             00001300   130              125
C        3 - PERCENT DATA ITEM VALIDITY ERROR                       00001310   131              126
C        4 - PERCENT DATA ITEM COMPATABILITY ERROR                  00001320   132              127
C        5 - PROCESS TIME FOR DIRECT MODE DATA ITEMS (SECONDS)      00001330   133              128
C        6 - PROCESS TIME FOR TUTORIAL MODE DATA ITEMS (SECONDS)    00001340   134              129
C        7 - PERCENT OF TRANSACTIONS CANCELLED                      00001350   135              130
C                                                                   00001360   136              131
C                                                                   00001370   137              132
       REAL TPARM(42)                                               00001380   138              133
C                                                                   00001390   139              134
       DATA TPARM                                                   00001400   140              135
C                  1    2     3     4     5     6     7             00001410   141              136
C          GRADE CHANGE                                             00001420   142              137
     I         / 60., 10.,  2.89,  6.54,  3.8, 32.3, 21.,           00001430   143              138
C          DUTY STATUS CHANGE                                       00001440   144              139

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                         INPUT CARD IMAGE                                      INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO REF-NO

     2          63., 8., 3.40, 6.40, 11.1, 26.4, 7.,                  00001450   145                140
C         INQUIRY                                                     00001460   146                141
     3          82., 3., 1.75, 10.53, 18.0, 27.7, 0.,                 00001470   147                142
C         ARRIVAL                                                     00001480   148                143
     4          59., 9., 2.00, 2.18, 16.9, 32.3, 16.,                 00001490   149                144
C         DEPARTURE                                                   00001500   150                145
     5          57., 12., 2.60, 4.36, 13.7, 27.9, 18.,                00001510   151                146
C         ADD INDIVIDUAL                                              00001520   152                147
     6           0., 70., 4.00, 1.43, 0.0, 20.5, 0. /                 00001530   153                148
C                                                                     00001540   154                149
      EQUIVALENCE ($SAVE(5),IP), ($SAVE(6),SEED), ($SAVE(7),SUB)      00001550   155                150
C                                                                     00001560   156                151
      END: DECLARATIONS;                                              00001570   157                152
C                                                                     00001580   158                153
      SEED = 1                                                        00001590   159                154
      DO 10 I = 1,10                                                  00001600   160                155
   10 TLOC(I) = 0                                                     00001610   161                156
C                                                                     00001620   162                157
      END: INITIALIZATION;                                            00001630   163                158
      SEIZE: FACILITY=TRANS;                                          00001640   164                159
C                                                                     00001650   165                160
C     SELECT TYPE OF TRANSACTION                                      00001660   166                161
      RN = SUNFM(0.0, 1.0, SEED)                                      00001670   167                162
      RP = SPWLIN(POINTS,7,RN)                                        00001680   168                163

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                      INPUT CARD IMAGE                                     INPUT    ALT INPUT  PAPER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO REF-NO

      IP = RP                                                         00001690   169                 164
C                                                                     00001700   170                 165
   50 SUB = (((IP-1)*7)+1                                             00001710   171                 166
      INVOKE: SERVICE = VALIDS, PARAMETER LIST=(SEED,SUB,TPARM);      00001720   172                 167
      WAIT RETURN: SERVICE=VALID;                                     00001730   173                 168
      IF(TLOC(8) .EQ. 1) GO TO 950                                    00001740   174                 169
C     TRANSACTION IS NOT CANCELLED                                    00001750   175                 170
      GO TO (100,200,300,400,500,600),IP                              00001760   176                 171
C                                                                     00001770   177                 172
C     START GRADE CHANGE PROCESSING                                   00001780   178                 173
  100 CONTINUE                                                        00001790   179                 174
      INVOKE: SERVICE=GRCHS, PARAMETER LIST=(TPARM(4),SEED);          00001800   180                 175
      WAIT RETURN: SERVICE=GRCH;                                      00001810   181                 176
      GO TO 900                                                       00001820   182                 177
C                                                                     00001830   183                 178
C                                                                     00001840   184                 179
C     START DUTY STATUS CHANGE PROCESSING                             00001850   185                 180
  200 CONTINUE                                                        00001860   186                 181
      INVOKE: SERVICE=DTYCS, PARAMETER LIST=(TPARM(11),SEED);         00001870   187                 182
      WAIT RETURN: SERVICE=DTYC;                                      00001880   188                 183
      GO TO 900                                                       00001890   189                 184
C                                                                     00001900   190                 185
C     START INQUIRY PROCESSING                                        00001910   191                 186
  300 CONTINUE                                                        00001920   192                 187

........10........20........30........40........50........60........70........80
```

```
                                INFORMATION PROCESSING SYSTEM SIMULATOR                                  PAGE    9

                                  STANDARD INPUT STREAM LISTING

                                       DATE ... 09/10/77

                             INPUT CARD IMAGE                                       INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

       INVOKE: SERVICE=INQS, PARAMETER LIST=(TPARM(18),SEED);           00001930     193                 188
       WAIT RETURN: SERVICE=INQ;                                        00001940     194                 189
       GO TO 900                                                        00001950     195                 190
C                                                                       00001960     196                 191
C      START ARRIVAL PROCESSING                                         00001970     197                 192
   400 CONTINUE                                                         00001980     198                 193
       INVOKE: SERVICE=ARRIVS, PARAMETER LIST=(TPARM(25),SEED);         00001990     199                 194
       WAIT RETURN: SERVICE=ARRIV;                                      00002000     200                 195
       GO TO 900                                                        00002010     201                 196
C                                                                       00002020     202                 197
C      START DEPARTURE PROCESSING                                       00002030     203                 198
   500 CONTINUE                                                         00002040     204                 199
       INVOKE: SERVICE=DEPTRS, PARAMETER LIST=(TPARM(32),SEED);         00002050     205                 200
       WAIT RETURN: SERVICE=DEPTR;                                      00002060     206                 201
       GO TO 900                                                        00002070     207                 202
C                                                                       00002080     208                 203
C      START ADDITION PROCESSING                                        00002090     209                 204
   600 CONTINUE                                                         00002100     210                 205
       INVOKE: SERVICE=ADDSLS, PARAMETER LIST=(TPARM(39),SEED);         00002110     211                 206
       WAIT RETURN: SERVICE=ADDSL;                                      00002120     212                 207
       GO TO 900                                                        00002130     213                 208
C                                                                       00002140     214                 209
   900 CONTINUE                                                         00002150     215                 210
C      CHECK FOR COMPATABILITY ERROR IN TRANSACTION                     00002160     216                 211

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                          PAGE   10
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                 INPUT CARD IMAGE                                               INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

      IF(IP .LT. 0) GO TO 950                                           00002170   217               212
      IF(IP .GT. 10) GO TO 950                                          00002180   218               213
      IF(YLOC(IP) .EQ. 0) GO TO 950                                     00002190   219               214
C     COMPATABILITY ERROR                                               00002200   220               215
      YLOC(IP) = 0                                                      00002210   221               216
      GO TO 50                                                          00002220   222               217
  950 CONTINUE                                                          00002230   223               218
      RELEASE: FACILITY=TRANS;                                          00002240   224               219
      END: ENDO SERVICE;                                                00002250   225               220
                                                                        00002260   226
                                                                        00002270   227
      ENDO SERVICE: ID=VALID, NAME=VALID#, SAVE AREA SIZE=10,           00002280   228               221
          PARAMETER LIST=(SEEDI,SUBI,TPARM);                            00002290   229               222
C     THIS SERVICE REPRESENTS THE TERMINAL INTERACTIONS INVOLVED        00002300   230               223
C     IN OBTAINING A TRANSACTION WITH NO VALIDATION ERRORS.             00002310   231               224
      COMMON /WK/ YLOC                                                  00002320   232               225
      INTEGER SEED,SUB,RETC,$UNFMI,SEEDI,SUBI,YLOC(10)                  00002330   233               226
      REAL DARR(42),DM,NUMDI,PVERR,PCERR,PYDM,PYTM,PCANCL,TPARM(42)     00002340   234               227
      REAL $UNFM                                                        00002350   235               228
      END: DECLARATIONS;                                                00002360   236               229
      DO 10 I = 1,42                                                    00002370   237               230
   10 DARR(I) = TPARM(I)                                                00002380   238               231
      SEED = SEEDI                                                      00002390   239               232
      SUB = SUBI                                                        00002400   240               233

........10........20........30........40........50........60........70........80
```

```
                              INFORMATION PROCESSING SYSTEM SIMULATOR                               PAGE   11
                               STANDARD INPUT STREAM LISTING
                                     DATE ... 09/10/77
                         INPUT CARD IMAGE                                           INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

      END: INITIALIZATION;                                               00002410    241                 234
      SEIZE: FACILITY=VALID;                                             00002420    242                 235
C                                                                        00002430    243                 236
C     GET PROBABILITY OF DIRECT MODE OF OPERATION                        00002440    244                 237
      DM = DARR(SUB+0)/100.                                              00002450    245                 238
C     GET NUMBER OF DATA ITEMS THIS TRANSACTION                          00002460    246                 239
      NUMDI = DARR(SUB+1)                                                00002470    247                 240
C     GET PROBABILITY OF VALIDITY ERROR                                  00002480    248                 241
      PVERR = DARR(SUB+2)/100.                                           00002490    249                 242
C     PVERR = 0.                                                         00002500    250                 243
C     GET PROBABILITY OF COMPATAPILITY ERROR                             00002510    251                 244
      PCERR = DARR(SUB+3)/100.                                           00002520    252                 245
C     PCERR = 0.                                                         00002530    253                 246
C     GET DIRECT MODE PROCESS TIME IN MSEC                               00002540    254                 247
      PTDM = DARR(SUB+4) * 1000.                                         00002550    255                 248
C     GET TUTORIAL MODE PROCESS TIME IN MSEC                             00002560    256                 249
      PTTM = DARR(SUB+5) * 1000.                                         00002570    257                 250
C     GET PROBABILITY THAT THIS TRANSACTION WILL BE CANCELLED            00002580    258                 251
      PCANCL = DARR(SUB+6)/100.                                          00002590    259                 252
C                                                                        00002600    260                 253
C     SELECT INPUT MODE (EITHER DIRECT OR TUTORIAL)                      00002610    261                 254
      RN = $UNFM(0.0, 1.0, SEED)                                         00002620    262                 255
      IF(RN .GT. DM) GO TO 50                                            00002630    263                 256
C     DIRECT MODE OF OPERATION                                           00002640    264                 257

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77


```
                         INPUT CARD IMAGE                                 INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80    SEQ-NO  LEV SEQ-NO REF-NO

C       WAIT FOR DATA TO BE ENTERED  (SYSTEM HOOK)                      00002650   265               258
        PROCESS: TIME=PTDM;                                             00002660   266               259
C       DETERMINE IF A VALIDITY ERROR HAS OCCURRED ON THE DATA ITEM     00002670   267               260
        RN = $UNFM(0.0, 1.0, SEED)                                      00002680   268               261
        IF(RN .GT. PVERR) GO TO 40                                      00002690   269               262
C       VALIDITY ERROR - WAIT FOR CORRECTED INPUT  (SYSTEM HOOK)        00002700   270               263
        PROCESS: TIME=PTDM;                                             00002710   271               264
C       NO VALIDITY ERROR                                               00002720   272               265
   40   CONTINUE                                                        00002730   273               266
        GO TO 100                                                       00002740   274               267
C                                                                       00002750   275               268
C                                                                       00002760   276               269
C     TUTORIAL MODE OF OPERATION                                        00002770   277               270
   50   J = 1                                                           00002780   278               271
   60   IF(J .GT. NUMDI) GO TO 100                                      00002790   279               272
C       WAIT FOR INPUT FROM TERMINAL  (SYSTEM HOOK)                     00002800   280               273
        PROCESS: TIME=PTTM;                                             00002810   281               274
C       DETERMINE IF VALIDITY ERROR HAS OCCURRED FOR THIS DATA ITEM     00002820   282               275
        RN = $UNFM(0.0, 1.0, SEED)                                      00002830   283               276
        IF(RN .GT. PVERR) GO TO 70                                      00002840   284               277
C       VALIDITY ERROR - WAIT FOR CORRECTED INPUT  (SYSTEM HOOK)        00002850   285               278
        PROCESS: TIME=PTTM;                                             00002860   286               279
   70   J = J + 1                                                       00002870   287               280
        GO TO 60                                                        00002880   288               281

........10........20........30........40........50........60........70........80
```

```
                           INFORMATION PROCESSING SYSTEM SIMULATOR                               PAGE   13
                              STANDARD INPUT STREAM LISTING
                                     DATE ... 09/10/77
                        INPUT CARD IMAGE                                     INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

C                                                                 00002890      289                  282
C     DETERMINE IF TRANSACTION IS TO BE CANCELLED                 00002900      290                  283
C       IF CANCELLED THEN RETC=1, OTHERWISE RETC=0                00002910      291                  284
  100 RN = $UNFM(0.0, 1.0, SEED)                                  00002920      292                  285
      TLOC(8) = 0                                                 00002930      293                  286
      IF(RN .LT. PCANCL) TLOC(8) = 1                              00002940      294                  287
      RELEASE: FACILITY=VALID;                                    00002950      295                  288
      END: ENDO SERVICE;                                          00002960      296                  289
                                                                  00002970      297
                                                                  00002980      298
                                                                  00002990      299
      ENDO SERVICE: ID=GRCH, NAME=GRCH$, SAVE AREA SIZE=10,       00003000      300                  290
                   PARAMETER LIST=(PERRI,SEEDI);                  00003010      301                  291
      COMMON /WK/ TLOC                                            00003020      302                  292
      INTEGER $UNFMI,SEED,SEEDI,TLOC(10)                          00003030      303                  293
      INTEGER OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS             00003040      304                  294
      INTEGER DMY,NEXT,PRIOR,OWNER                                00003050      305                  295
      INTEGER ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA  00003060      306                  296
      INTEGER CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI 00003070  307                  297
      EQUIVALENCE ($SAVE(1),PERRI, ($SAVE(2),SEED)                00003080      308                  298
      REAL $UNFM,RN,PERR,PERRI                                    00003090      309                  299
      DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/ 00003100      310                  300
      DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                          00003110      311                  301
      DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA     00003120      312                  302

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

INPUT CARD IMAGE ....10........20........30........40........50........60........70	80	INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	MEMBER REF-NO
I / 1, 2, 3, 4, 5, 6, 7, 8, 9, 10/	00003130	313		303
DATA CALC,GRUI,IDUI,INUM,PMID,PCUI,SMYD,UMAW,UMGR,UMPOS,UMUI	00003140	314		304
I / 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11/	00003150	315		305
END: DECLARATIONS;	00003160	316		306
PERR = PERRI/100.	00003170	317		307
SEED = SEEDI	00003180	318		308
END: INITIALIZATION;	00003190	319		309
SEIZE: FACILITY=GRCH;	00003200	320		310
C RETURN CODE -TLOC(I)- = 0 IF NO ERROR, 1 IF ERROR	00003210	321		311
C PARAMETER LIST CONTAINS-	00003220	322		312
C 1) DML TYPE - OBTAIN=1,FIND=2,MODIFY=3,STORE=4,CONNECT=5,	00003230	323		313
C DISCONNECT=6,ERASE=7	00003240	324		314
C 2) QUALIFIER - DMY=0,NEXT=1,PRIOR=2,OWNER=3	00003250	325		315
C 2) RECORD TYPE - IDENTIFIER OF THE RECORD UPON WHICH THE DML IS	00003260	326		316
C TO OPERATE	00003270	327		317
C 4) SET IDENTIFIER - INDEX	00003280	328		318
C	00003290	329		319
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,IDEN,CALC);	00003300	330		320
WAIT RETURN: SERVICE=INTF;	00003310	331		321
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADDI,CALC);	00003320	332		322
WAIT RETURN: SERVICE=INTF;	00003330	333		323
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,MOSC,CALC);	00003340	334		324
WAIT RETURN: SERVICE=INTF;	00003350	335		325
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);	00003360	336		326

........10........20........30........40........50........60........70........80

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                   PAGE   15

                                STANDARD INPUT STREAM LISTING

                                     DATE ... 09/10/77

                         INPUT CARD IMAGE                                          INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

      WAIT RETURN: SERVICE=INTF;                                          00003370   337                 327
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,OWNER,DMY,UMUI);      00003380   338                 328
      WAIT RETURN: SERVICE=INTF;                                          00003390   339                 329
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,POSN,CALC);       00003400   340                 330
      WAIT RETURN: SERVICE=INTF;                                          00003410   341                 331
      RN = $UNFM(0.0, 1.0, SEED)                                          00003420   342                 332
      IF(RN .LT. PERR) GO TO 100                                          00003430   343                 333
C                                                                         00003440   344                 334
C     END COMPATABILITY CHECK - NO COMPATABILITY ERROR DETECTED           00003450   345                 335
C     MODIF IDENT AND ADD-IDENT RECORDS                                   00003460   346                 336
C                                                                         00003470   347                 337
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,ADDI,CALC);       00003480   348                 338
      WAIT RETURN: SERVICE=INTF;                                          00003490   349                 339
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,ADDI,CALC);       00003500   350                 340
      WAIT RETURN: SERVICE=INTF;                                          00003510   351                 341
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC);       00003520   352                 342
      WAIT RETURN: SERVICE=INTF;                                          00003530   353                 343
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,IDEN,DMY);        00003540   354                 344
      WAIT RETURN: SERVICE=INTF;                                          00003550   355                 345
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC);       00003560   356                 346
      WAIT RETURN: SERVICE=INTF;                                          00003570   357                 347
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(DISCN,DMY,IDEN,PMID);       00003580   358                 348
      WAIT RETURN: SERVICE=INTF;                                          00003590   359                 349
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,MOSC,DMY);        00003600   360                 350

........10........20........30........40........50........60........70........80
```

INFORMATION PROCESSING SYSTEM SIMULATOR

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

INPUT CARD IMAGE ....10........20........30........40........50........60........70........80		INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	MEMBER REF-NO
WAIT RETURN: SERVICE=INTF;	00003610	361		351
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC);	00003620	362		352
WAIT RETURN: SERVICE=INTF;	00003630	363		353
INVOKE: SERVICE= INTFS, PARAMETER LIST=(CONCT,DMY,IDEN,PMID);	00003640	364		354
WAIT RETURN: SERVICE=INTF;	00003650	365		355
C	00003660	366		356
C DISCNNECT THE PERSON'S UIST RECORD	00003670	367		357
C	00003680	368		358
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);	00003690	369		359
WAIT RETURN: SERVICE=INTF;	00003700	370		360
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,OWNER,DMY,GRUI);	00003710	371		361
WAIT RETURN: SERVICE=INTF;	00003720	372		362
INVOKE: SERVICE= INTFS, PARAMETER LIST=(DISCN,DMY,UIST,GRUI);	00003730	373		363
WAIT RETURN: SERVICE=INTF;	00003740	374		364
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);	00003750	375		365
WAIT RETURN: SERVICE=INTF;	00003760	376		366
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,OWNER,UIST,GRUI);	00003770	377		367
WAIT RETURN: SERVICE=INTF;	00003780	378		368
INVOKE: SERVICE= INTFS, PARAMETER LIST=(DISCN,DMY,UIST,GRUI);	00003790	379		369
WAIT RETURN: SERVICE=INTF;	00003800	380		370
C	00003810	381		371
C CONNECT UIST RECORDS TO THE GRADE-UIST SET OF THE NEW GRADE	00003820	382		372
C	00003830	383		373
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);	00003840	384		374

........10........20........30........40........50........60........70........80

```
                    INFORMATION PROCESSING SYSTEM SIMULATOR                                    PAGE  17
                          STANDARD INPUT STREAM LISTING
                              DATE ... 09/10/77
                     INPUT CARD IMAGE                                    INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

        WAIT RETURN: SERVICE=INTF;                                       00003850   385              375
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC);    00003860   386              376
        WAIT RETURN: SERVICE=INTF;                                       00003870   387              377
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,NEXT,GRADE,UMGR);  00003880   388              378
        WAIT RETURN: SERVICE=INTF;                                       00003890   389              379
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,UIST,CALC);    00003900   390              380
        WAIT RETURN: SERVICE=INTF;                                       00003910   391              381
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,OWNER,DMY,GRUI);   00003920   392              382
        WAIT RETURN: SERVICE=INTF;                                       00003930   393              383
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(CONCT,DMY,UIST,GRUI);    00003940   394              384
        WAIT RETURN: SERVICE=INTF;                                       00003950   395              385
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);   00003960   396              386
        WAIT RETURN: SERVICE=INTF;                                       00003970   397              387
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(CONCT,DMY,UIST,GRUI);    00003980   398              388
        WAIT RETURN: SERVICE=INTF;                                       00003990   399              389
C       NORMAL RETURN                                                    00004000   400              390
     90 TLOC(I) = 0                                                      00004010   401              391
        GO TO 150                                                        00004020   402              392
C       COMPATABILITY ERROR                                              00004030   403              393
    100 CONTINUE                                                         00004040   404              394
        TLOC(I) = 1                                                      00004050   405              395
    150 CONTINUE                                                         00004060   406              396
        RELEASE: FACILITY=GRCH;                                          00004070   407              397
        END: ENDO SERVICE;                                               00004080   408              398

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                          PAGE  18
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                       INPUT CARD IMAGE                                    INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

                                                                  00004090   409
                                                                  00004100   410
                                                                  00004110   411
   ENDO SERVICE: ID=DTYC, NAME=DTYC$, SAVE AREA SIZE=10,          00004120   412              399
                 PARAMETER LIST=(PERRI,SEEDI);                    00004130   413              400
   COMMON /WK/ TLOC                                               00004140   414              401
   INTEGER $UNFMI,SEED,SEEDI,TLOC(10)                             00004150   415              402
   INTEGER OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS                00004160   416              403
   INTEGER DMY,NEXT,PRIOR,OWNER                                   00004170   417              404
   INTEGER ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA     00004180   418              405
   INTEGER CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI 00004190  419              406
   EQUIVALENCE ($SAVE(1),PERR), ($SAVE(2),SEED)                   00004200   420              407
   REAL $UNFM,RN,PERR,PERRI                                       00004210   421              408
   DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/    00004220   422              409
   DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                             00004230   423              410
   DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA        00004240   424              411
  1 /   1,   2,   3,    4,   5,   6,   7,   8,   9,  10/          00004250   425              412
   DATA CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI   00004260   426              413
  1 /   1,   2,   3,   4,   5,   6,   7,   8,   9,   10,  11/     00004270   427              414
   END: DECLARATIONS;                                             00004280   428              415
   PERR = PERRI/100.                                              00004290   429              416
   SEED = SEEDI                                                   00004300   430              417
   END: INITIALIZATION;                                           00004310   431              418
   SEIZE: FACILITY=DTYC;                                          00004320   432              419

........10........20........30........40........50........60........70........80
```

INFORMATION PROCESSING SYSTEM SIMULATOR

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

INPUT CARD IMAGE ....10........20........30........40........50........60........70........80		INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	REFERENCE REF-NO
C	00004330	433		420
C RETURN CODE -TLOC(2)- = 0 IF NO ERROR, 1 IF ERROR	00004340	434		421
C	00004350	435		422
C PARAMETER LIST CONTAINS-	00004360	436		423
C 1) DML TYPE - OBTAIN=1,FIND=2,MODIFY=3,STORE=4,CONNECT=5,	00004370	437		424
C DISCONNECT=6,ERASE=7	00004380	438		425
C 2) QUALIFIER - DMY=0,NEXT=1,PRIOR=2,OWNER=3	00004390	439		426
C 2) RECORD TYPE - IDENTIFIER OF THE RECORD UPON WHICH THE DML IS	00004400	440		427
C TO OPERATE	00004410	441		428
C 4) SET IDENTIFIER - INDEX	00004420	442		429
C	00004430	443		430
C COMPATABILITY CHECK	00004440	444		431
C	00004450	445		432
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,IDEN,CALC);	00004460	446		433
WAIT RETURN: SERVICE=INTF;	00004470	447		434
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADDI,CALC);	00004480	448		435
WAIT RETURN: SERVICE=INTF;	00004490	449		436
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);	00004500	450		437
WAIT RETURN: SERVICE=INTF;	00004510	451		438
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);	00004520	452		439
WAIT RETURN: SERVICE=INTF;	00004530	453		440
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,OWNER,DMY,UMUI);	00004540	454		441
WAIT RETURN: SERVICE=INTF;	00004550	455		442
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADUN,CALC);	00004560	456		443

........10........20........30........40........50........60........70........80

INPUT CARD IMAGE ....10........20........30........40........50........60........70........80		INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	MEMBER REF-NO
WAIT RETURN: SERVICE=INTF;	00004570	457		444
C DETERMINE IF COMPATABILITY ERROR HAS OCCURRED	00004580	458		445
RN = $UNFM(0.0, 1.0, SEED)	00004590	459		446
IF(RN .LT. FERR) GO TO 100	00004600	460		447
C NO COMPATABILITY ERROR	00004610	461		448
C	00004620	462		449
C MODIFY THE IDENT, UIST AND ADD-UNIT RECORDS	00004630	463		450
C	00004640	464		451
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC);	00004650	465		452
WAIT RETURN: SERVICE=INTF;	00004660	466		453
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,IDEN,DMY);	00004670	467		454
WAIT RETURN: SERVICE=INTF;	00004680	468		455
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,UIST,CALC);	00004690	469		456
WAIT RETURN: SERVICE=INTF;	00004700	470		457
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,UIST,DMY);	00004710	471		458
WAIT RETURN: SERVICE=INTF;	00004720	472		459
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,ADUN,CALC);	00004730	473		460
WAIT RETURN: SERVICE=INTF;	00004740	474		461
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,ADUN,DMY);	00004750	475		462
WAIT RETURN: SERVICE=INTF;	00004760	476		463
C NORMAL RETURN	00004770	477		464
90 TLOC(2) = 0	00004780	478		465
GO TO 150	00004790	479		466
C COMPATABILITY ERROR	00004800	480		467

........10........20........30........40........50........60........70........80

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                              PAGE  21
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                        INPUT CARD IMAGE                                        INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

   100 TLOC(2) = 1                                                   00004810    481                 468
   150 CONTINUE                                                      00004820    482                 469
       RELEASE: FACILITY=DTYC;                                       00004830    483                 470
       END: ENDO SERVICE;                                            00004840    484                 471
                                                                     00004850    485
                                                                     00004860    486
                                                                     00004870    487
       ENDO SERVICE: ID=INQ, NAME=INQ$, SAVE AREA SIZE=10,           00004880    488                 472
                     PARAMETER LIST=(PERRT,SEEDI);                   00004890    489                 473
       COMMON /WK/ TLOC                                              00004900    490                 474
       INTEGER $UNFMI,SEED,SEEDI,TLOC(10)                            00004910    491                 475
       INTEGER OBTAN,FINDR,MODIF,STOP,CONCT,DISCN,ERAS               00004920    492                 476
       INTEGER DMY,NEXT,PRIOR,OWNER                                  00004930    493                 477
       INTEGER ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA    00004940    494                 478
       INTEGER CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI 00004950  495                 479
       EQUIVALENCE ($SAVE(1),PERR), ($SAVE(2),SEED)                  00004960    496                 480
       REAL $UNFM,RN,PERR,PERRI                                      00004970    497                 481
       DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/   00004980    498                 482
       DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                            00004990    499                 483
       DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA       00005000    500                 484
      1  /    1,   2,   3,    4,   5,   6,   7,   8,   9,  10/       00005010    501                 485
       DATA CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI  00005020    502                 486
      1  /    1,   2,   3,   4,   5,   6,   7,   8,   9,   10,  11/  00005030    503                 487
       END: DECLARATIONS;                                            00005040    504                 488

........10........20........30........40........50........60........70........80
```

```
                              INFORMATION PROCESSING SYSTEM SIMULATOR                                           PAGE   22
                                 STANDARD INPUT STREAM LISTING
                                      DATE ... 09/10/77
                               INPUT CARD IMAGE                                             INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80           SEQ-NO  LEV SEQ-NO   REF-NO

      PERR = PERRI/100.                                                        00005050      505                   489
      SEED = SEEDI                                                             00005060      506                   490
      END: INITIALIZATION;                                                     00005070      507                   491
      SEIZE: FACILITY=INQ;                                                     00005080      508                   492
C                                                                              00005090      509                   493
C     RETURN CODE -TLOC(3)- = 0 IF NO ERROR, 1 IF ERROR                        00005100      510                   494
C                                                                              00005110      511                   495
C     PARAMETER LIST CONTAINS-                                                 00005120      512                   496
C       1) DML TYPE   - OBTAIN=1,FIND=2,MODIFY=3,STORE=4,CONNECT=5,            00005130      513                   497
C                       DISCONNECT=6,ERASE=7                                   00005140      514                   498
C       2) QUALIFIER - DMY=0,NEXT=1,PRIOR=2,OWNER=3                            00005150      515                   499
C       2) RECORD TYPE - IDENTIFIER OF THE RECORD UPON WHICH THE DML IS        00005160      516                   500
C                         TO OPERATE                                           00005170      517                   501
C       4) SET IDENTIFIER - INDEX                                              00005180      518                   502
C                                                                              00005190      519                   503
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,IDEN,CALC);            00005200      520                   504
      WAIT RETURN: SERVICE=INTF;                                               00005210      521                   505
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,INST,CALC);            00005220      522                   506
      WAIT RETURN: SERVICE=INTF;                                               00005230      523                   507
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADDI,CALC);            00005240      524                   508
      WAIT RETURN: SERVICE=INTF;                                               00005250      525                   509
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);           00005260      526                   510
      WAIT RETURN: SERVICE=INTF;                                               00005270      527                   511
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,POSN,CALC);            00005280      528                   512

........10........20........30........40........50........60........70........80
```

```
                                INFORMATION PROCESSING SYSTEM SIMULATOR                                    PAGE  23
                                   STANDARD INPUT STREAM LISTING
                                        DATE ... 09/10/77
                     INPUT CARD IMAGE                                             INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO REF-NO

      WAIT RETURN: SERVICE=INTF;                                          00005290   529              513
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ALOC,CALC);       00005300   530              514
      WAIT RETURN: SERVICE=INTF;                                          00005310   531              515
C     DETERMINE IF COMPATABILITY ERROR HAS OCCURRED                       00005320   532              516
      RN = SUNFM(0.0, 1.0, SEED)                                          00005330   533              517
      IF(PERR .LT. RN) GO TO 100                                          00005340   534              518
C     NORMAL RETURN                                                       00005350   535              519
   90 TLOC(3) = 0                                                         00005360   536              520
      GO TO 150                                                           00005370   537              521
C     COMPATABILITY ERROR                                                 00005380   538              522
  100 TLOC(3) = 1                                                         00005390   539              523
  150 CONTINUE                                                            00005400   540              524
      RELEASE: FACILITY=INQ;                                              00005410   541              525
      END: ENDO SERVICE;                                                  00005420   542              526
                                                                          00005430   543
                                                                          00005440   544
                                                                          00005450   545
      ENDO SERVICE: ID=ARRIV, NAME=ARRIVS, SAVE AREA SIZE=10,             00005460   546              527
                   PARAMETER LIST=(PERRI,SEEDI);                          00005470   547              528
      COMMON /WK/ TLOC                                                    00005480   548              529
      INTEGER SUNFMI,SEED,SEEDI,TLOC(10)                                  00005490   549              530
      INTEGER OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS                     00005500   550              531
      INTEGER DMY,NEXT,PRIOR,OWNER                                        00005510   551              532
      INTEGER ADDI,ADIW,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA          00005520   552              533

........10........20........30........40........50........60........70........80
```

```
                              INFORMATION PROCESSING SYSTEM SIMULATOR                                  PAGE   74
                                 STANDARD INPUT STREAM LISTING
                                       DATE ... 09/10/77
                     INPUT CARD IMAGE                                                   INPUT    ALT INPUT  MERGER
........10........20........30........40........50........60........70........80       SEQ-NO   LEV SEQ-NO REF-NO

      INTEGER CALC,GRUI,IDUI,INUM,PHYD,POUI,SMID,UMAW,UMGR,UMPOS,UMUI       00005530     553                 534
      EQUIVALENCE ($SAVE(1),PERR), ($SAVE(2),SEED)                          00005540     554                 535
      REAL SUNFM,RN,PERR,PERRI                                              00005550     555                 536
      DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/           00005560     556                 537
      DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                                    00005570     557                 538
      DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA               00005580     558                 539
     1  /   1,   2,   3,    4,   5,   6,   7,   8,   9,  10/                00005590     559                 540
      DATA CALC,GRUI,IDUI,INUM,PHYD,POUI,SMID,UMAW,UMGR,UMPOS,UMUI          00005600     560                 541
     1  /   1,   2,   3,   4,   5,   6,   7,   8,   9,   10,  11/           00005610     561                 542
      END: DECLARATIONS;                                                    00005620     562                 543
      PERR = PERRI/100.                                                     00005630     563                 544
      SEED = SEEDI                                                          00005640     564                 545
      END: INITIALIZATION;                                                  00005650     565                 546
      SEIZE: FACILITY=ARRIV;                                                00005660     566                 547
C                                                                           00005670     567                 548
C     RETURN CODE -TLOC(4)- = 0 IF NO ERROR, 1 IF ERROR                     00005680     568                 549
C                                                                           00005690     569                 550
C     PARAMETER LIST CONTAINS-                                              00005700     570                 551
C      1) DML TYPE  - OBTAIN=1,FIND=2,MODIFY=3,STORE=4,CONNECT=5,           00005710     571                 552
C                     DISCONNECT=6,ERASE=7                                  00005720     572                 553
C      2) QUALIFIER - DMY=0,NEXT=1,PRIOR=2,OWNER=3                          00005730     573                 554
C      2) RECORD TYPE - IDENTIFIER OF THE RECORD UPON WHICH THE DML IS      00005740     574                 555
C                       TO OPERATE                                          00005750     575                 556
C      4) SET IDENTIFIER - INDEX                                            00005760     576                 557

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                              PAGE   25
                            STANDARD INPUT STREAM LISTING
                                 DATE ... 09/10/77
                      INPUT CARD IMAGE                                         INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80     SEQ-NO   LEV SEQ-NO  REF-NO

C                                                                 00005770      577                  558
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,IDEN,CALC);     00005780      578                  559
     WAIT RETURN: SERVICE=INTF;                                   00005790      579                  560
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,INST,CALC);     00005800      580                  561
     WAIT RETURN: SERVICE=INTF;                                   00005810      581                  562
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,ADDI,CALC);     00005820      582                  563
     WAIT RETURN: SERVICE=INTF;                                   00005830      583                  564
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,POSN,CALC);     00005840      584                  565
     WAIT RETURN: SERVICE=INTF;                                   00005850      585                  566
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);    00005860      586                  567
     WAIT RETURN: SERVICE=INTF;                                   00005870      587                  568
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);    00005880      588                  569
     WAIT RETURN: SERVICE=INTF;                                   00005890      589                  570
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,UNMA,CALC);     00005900      590                  571
     WAIT RETURN: SERVICE=INTF;                                   00005910      591                  572
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,ADUN,CALC);     00005920      592                  573
     WAIT RETURN: SERVICE=INTF;                                   00005930      593                  574
     INVOKE: SERVICE= INTF$, PARAMETER LIST=(OBTAN,DMY,ALOC,CALC);     00005940      594                  575
     WAIT RETURN: SERVICE=INTF;                                   00005950      595                  576
     RN = $UNFM(0.0, 1.0, SEED)                                   00005960      596                  577
     IF(RN .LT. PERR) GO TO 100                                   00005970      597                  578
C                                                                 00005980      598                  579
C    END COMPATABILITY CHECK - NO COMPATABILITY ERROR DETECTED    00005990      599                  580
C    MODIFY IDENT AND ADD-IDENT RECORDS                           00006000      600                  581

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

INPUT CARD IMAGE ....10........20........30........40........50........60........70........80	INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	MEMBER REF-NO
C 00006010	601		582
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ALOC,CALC); 00006020	602		583
WAIT RETURN: SERVICE=INTF; 00006030	603		584
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC); 00006040	604		585
WAIT RETURN: SERVICE=INTF; 00006050	605		586
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,IDEN,DMY); 00006060	606		587
WAIT RETURN: SERVICE=INTF; 00006070	607		588
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADDI,CALC); 00006080	608		589
WAIT RETURN: SERVICE=INTF; 00006090	609		590
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,ADDI,DMY); 00006100	610		591
WAIT RETURN: SERVICE=INTF; 00006110	611		592
C 00006120	612		593
C STORE THE NEW UIST RECORD AND ITS CONNECTION IN THE IDENT-UIST 00006130	613		594
C SET AND IN THE UNITMIT-UIST SET 00006140	614		595
C 00006150	615		596
INVOKE: SERVICE= INTFS, PARAMETER LIST=(STOR,DMY,UIST,DMY); 00006160	616		597
WAIT RETURN: SERVICE=INTF; 00006170	617		598
INVOKE: SERVICE= INTFS, PARAMETER LIST=(GETAN,DMY,GRADE,UMGR); 00006180	618		599
WAIT RETURN: SERVICE=INTF; 00006190	619		600
INVOKE: SERVICE= INTFS, PARAMETER LIST=(CONCT,DMY,UIST,POUI); 00006200	620		601
WAIT RETURN: SERVICE=INTF; 00006210	621		602
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,UNMA,CALC); 00006220	622		603
WAIT RETURN: SERVICE=INTF; 00006230	623		604
INVOKE: SERVICE= INTFS, PARAMETER LIST=(CONCT,DMY,UIST,UMUI); 00006240	624		605

........10........20........30........40........50........60........70........80

```
                    INFORMATION PROCESSING SYSTEM SIMULATOR                                       PAGE  27
                         STANDARD INPUT STREAM LISTING
                              DATE ... 09/10/77
                           INPUT CARD IMAGE                                     INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

         WAIT RETURN: SERVICE=INTF;                                    00006250   625                 606
         INVOKE: SERVICE= INTFS, PARAMETER LIST=(CONCT,DMY,UIST,GRUI); 00006260   626                 607
         WAIT RETURN: SERVICE=INTF;                                    00006270   627                 608
         RN = SUNFM(0.0, 1.0, SEED)                                    00006280   628                 609
         IF(RN .GT. 0.20) GO TO 100                                    00006290   629                 610
         INVOKE: SERVICE= INTFS, PARAMETER LIST=(DISCN,DMY,UIST,POUI); 00006300   630                 611
         WAIT RETURN: SERVICE=INTF;                                    00006310   631                 612
         INVOKE: SERVICE= INTFS, PARAMETER LIST=(DISCN,DMY,UIST,UMUI); 00006320   632                 613
         WAIT RETURN: SERVICE=INTF;                                    00006330   633                 614
         INVOKE: SERVICE= INTFS, PARAMETER LIST=(DISCN,DMY,UIST,GRUI); 00006340   634                 615
         WAIT RETURN: SERVICE=INTF;                                    00006350   635                 616
         INVOKE: SERVICE= INTFS, PARAMETER LIST=(ERAS,DMY,UIST,DMY);   00006360   636                 617
         WAIT RETURN: SERVICE=INTF;                                    00006370   637                 618
C        NORMAL RETURN                                                 00006380   638                 619
      90 TLOC(4) = 0                                                   00006390   639                 620
         GO TO 150                                                     00006400   640                 621
C        COMPATABILITY ERROR                                           00006410   641                 622
     100 CONTINUE                                                      00006420   642                 623
         TLOC(4) = 1                                                   00006430   643                 624
     150 CONTINUE                                                      00006440   644                 625
         RELEASE: FACILITY=ARRIV;                                      00006450   645                 626
         END: ENDO SERVICE;                                            00006460   646                 627
                                                                       00006470   647
                                                                       00006480   648

........10........20........30........40........50........60........70........80
```

```
                    INFORMATION PROCESSING SYSTEM SIMULATOR                                          PAGE   28
                          STANDARD INPUT STREAM LISTING
                               DATE ... 09/10/77
                     INPUT CARD IMAGE                                         INPUT    ALT INPUT    MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO   REF-NO

                                                                   00006490     649
    ENDO SERVICE: ID=DEPTR, NAME=DEPTRS, SAVE AREA SIZE=10,        00006500     650                 628
                  PARAMETER LIST=(PERRI,SEEDI);                    00006510     651                 629
    COMMON /WK/ TLOC                                               00006520     652                 630
    INTEGER $UNFMI,SEED,SEEDI,TLOC(10)                             00006530     653                 631
    INTEGER OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS                00006540     654                 632
    INTEGER DMY,NEXT,PRIOR,OWNER                                   00006550     655                 633
    INTEGER ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA     00006560     656                 634
    INTEGER CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI 00006570    657                 635
    EQUIVALENCE ($SAVE(1),PERRI), ($SAVE(2),SEEDI)                 00006580     658                 636
    REAL $UNFM,RN,PERR,PERRI                                       00006590     659                 637
    DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/    00006600     660                 638
    DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                             00006610     661                 639
    DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA        00006620     662                 640
   I  /   1,   2,   3,   4,   5,   6,   7,   8,   9,  10/          00006630     663                 641
    DATA CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI   00006640     664                 642
   I  /   1,   2,   3,   4,   5,   6,   7,   8,   9,   10,   11/    00006650     665                 643
    END: DECLARATIONS;                                             00006660     666                 644
    PERR = PERRI/100.                                              00006670     667                 645
    SEED = SEEDI                                                   00006680     668                 646
    END: INITIALIZATION;                                           00006690     669                 647
    SEIZE: FACILITY=DEPTR;                                         00006700     670                 648
C                                                                  00006710     671                 649
C   RETURN CODE -TLOC(5)- = 0 IF NO ERROR, 1 IF ERROR              00006720     672                 650

........10........20........30........40........50........60........70........80
```

INFORMATION PROCESSING SYSTEM SIMULATOR — PAGE 29

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

INPUT CARD IMAGE ....10........20........30........40........50........60........70........80		INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	MERGED REF-NO
C	00006730	673		651
C PARAMETER LIST CONTAINS-	00006740	674		652
C 1) DML TYPE - OBTAIN=1,FIND=2,MODIFY=3,STORE=4,CONNECT=5,	00006750	675		653
C DISCONNECT=6,ERASE=7	00006760	676		654
C 2) QUALIFIER - DMY=0,NEXT=1,PRIOR=2,OWNER=3	00006770	677		655
C 2) RECORD TYPE - IDENTIFIER OF THE RECORD UPON WHICH THE DML IS	00006780	678		656
C TO OPERATE	00006790	679		657
C 4) SET IDENTIFIER - INDEX	00006800	680		658
C	00006810	681		659
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,IDEN,CALC);	00006820	682		660
WAIT RETURN: SERVICE=INTF;	00006830	683		661
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,INST,CALC);	00006840	684		662
WAIT RETURN: SERVICE=INTF;	00006850	685		663
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADDI,CALC);	00006860	686		664
WAIT RETURN: SERVICE=INTF;	00006870	687		665
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,UNMA,CALC);	00006880	688		666
WAIT RETURN: SERVICE=INTF;	00006890	689		667
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADUN,CALC);	00006900	690		668
WAIT RETURN: SERVICE=INTF;	00006910	691		669
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,UNMA,CALC);	00006920	692		670
WAIT RETURN: SERVICE=INTF;	00006930	693		671
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ADUN,CALC);	00006940	694		672
WAIT RETURN: SERVICE=INTF;	00006950	695		673
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,ALOC,CALC);	00006960	696		674

........10........20........30........40........50........60........70........80

INFORMATION PROCESSING SYSTEM SIMULATOR PAGE 30

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

INPUT CARD IMAGE ....10........20........30........40........50........60........70........80		INPUT SEQ-NO	ALT INPUT LEV SEQ-NO	MEMBER REF-NO
WAIT RETURN: SERVICE=INTF;	00006970	697		675
C DETERMINE IF COMPATABILITY ERROR HAS OCCURRED	00006980	698		676
RN = $UNFM(0.0, 1.0, SEED)	00006990	699		677
IF(RN .LT. PERR) GO TO 100	00007000	700		678
C NO COMPATABILITY ERROR	00007010	701		679
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);	00007020	702		680
WAIT RETURN: SERVICE=INTF;	00007030	703		681
INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,NEXT,UIST,IDUI);	00007040	704		682
WAIT RETURN: SERVICE=INTF;	00007050	705		683
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,IDEN,CALC);	00007060	706		684
WAIT RETURN: SERVICE=INTF;	00007070	707		685
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,IDEN,DMY);	00007080	708		686
WAIT RETURN: SERVICE=INTF;	00007090	709		687
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);	00007100	710		688
WAIT RETURN: SERVICE=INTF;	00007110	711		689
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);	00007120	712		690
WAIT RETURN: SERVICE=INTF;	00007130	713		691
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,UIST,DMY);	00007140	714		692
WAIT RETURN: SERVICE=INTF;	00007150	715		693
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,UIST,DMY);	00007160	716		694
WAIT RETURN: SERVICE=INTF;	00007170	717		695
INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,ADDI,CALC);	00007180	718		696
WAIT RETURN: SERVICE=INTF;	00007190	719		697
INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,ADUN,DMY);	00007200	720		698

........10........20........30........40........50........60........70........80

```
                              INFORMATION PROCESSING SYSTEM SIMULATOR                                  PAGE  31
                                  STANDARD INPUT STREAM LISTING
                                      DATE ... 09/10/77
                   INPUT CARD IMAGE                                        INPUT   ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

      WAIT RETURN: SERVICE=INTF;                                          00007210   721               699
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,DMY,ADUN,CALC);       00007220   722               700
      WAIT RETURN: SERVICE=INTF;                                          00007230   723               701
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(MODIF,DMY,ADUN,DMY);        00007240   724               702
      WAIT RETURN: SERVICE=INTF;                                          00007250   725               703
      INVOKE: SERVICE= INTFS, PARAMETER LIST=(FINDR,NEXT,UIST,IDUI);      00007260   726               704
      WAIT RETURN: SERVICE=INTF;                                          00007270   727               705
C     NORMAL RETURN                                                       00007280   728               706
   90 TLOC(5) = 0                                                         00007290   729               707
      GO TO 150                                                           00007300   730               708
C     COMPATABILITY ERROR                                                 00007310   731               709
  100 CONTINUE                                                            00007320   732               710
      TLOC(5) = 1                                                         00007330   733               711
  150 CONTINUE                                                            00007340   734               712
      RELEASE: FACILITY=DEPTR;                                            00007350   735               713
      END: ENDO SERVICE;                                                  00007360   736               714
                                                                          00007370   737
                                                                          00007380   738
                                                                          00007390   739
      ENDO SERVICE: ID=ADDSL, NAME=ADDSL$, SAVE AREA SIZE=10,             00007400   740               715
                    PARAMETER LIST=(PERR1,SEED1);                         00007410   741               716
      COMMON /WK/ TLOC                                                    00007420   742               717
      INTEGER $UNFMI,SEED,SEED1,TLOC(10)                                  00007430   743               718
      INTEGER OBTAN,FINDR,MODIF,STOP,CONCT,DISCN,ERAS                     00007440   744               719

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                            PAGE   [illegible]2
                             STANDARD INPUT STREAM LISTING
                                 DATE ... 09/10/77
                  INPUT CARD IMAGE                                          INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

      INTEGER DMY,NEXT,PRIOR,OWNER                                          00007450   745              720
      INTEGER ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA            00007460   746              721
      INTEGER CALC,GRUI,IDUI,INUM,PHYD,POUI,SMID,UMAW,UMGR,UMPOS,UMUI       00007470   747              722
      EQUIVALENCE ($SAVE(1),PERR), ($SAVE(2),SEED)                          00007480   748              723
      REAL $UNFM,RN,PERR,PERRI                                              00007490   749              724
      DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/           00007500   750              725
      DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                                    00007510   751              726
      DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA               00007520   752              727
     1 /    1,   2,   3,    4,   5,   6,   7,   8,   9,  10/                00007530   753              728
      DATA CALC,GRUI,IDUI,INUM,PHYD,POUI,SMID,UMAW,UMGR,UMPOS,UMUI          00007540   754              729
     1 /    1,   2,   3,   4,   5,   6,   7,   8,   9,   10,  11/           00007550   755              730
      END: DECLARATIONS;                                                    00007560   756              731
      PERR = PERRI                                                          00007570   757              732
      SEED = SEEDI                                                          00007580   758              733
      END: INITIALIZATION;                                                  00007590   759              734
      SEIZE: FACILITY=ADDSL;                                                00007600   760              735
C                                                                           00007610   761              736
C     RETURN CODE -TLOC(6)- = 0 IF NO ERROR, 1 IF ERROR                     00007620   762              737
C                                                                           00007630   763              738
C     PARAMETER LIST CONTAINS-                                              00007640   764              739
C      1) DML TYPE  - OBTAIN=1,FIND=2,MODIFY=3,STORE=4,CONNECT=5,           00007650   765              740
C                     DISCONNECT=6,ERASE=7                                  00007660   766              741
C      2) QUALIFIER - DMY=0,NEXT=1,PRIOR=2,OWNER=3                          00007670   767              742
C      2) RECORD TYPE - IDENTIFIER OF THE RECORD UPON WHICH THE DML IS      00007680   768              743

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                  PAGE   23
                              STANDARD INPUT STREAM LISTING
                                  DATE ... 09/10/77
                    INPUT CARD IMAGE                                          INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

C                    TO OPERATE                                       00007690    769                 744
C       4) SET IDENTIFIER - INDEX                                     00007700    770                 745
C                                                                     00007710    771                 746
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,IDEN,CALC); 00007720    772                 747
        WAIT RETURN: SERVICE=INTF;                                    00007730    773                 748
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,UNMA,CALC); 00007740    774                 749
        WAIT RETURN: SERVICE=INTF;                                    00007750    775                 750
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,MOSC,CALC); 00007760    776                 751
        WAIT RETURN: SERVICE=INTF;                                    00007770    777                 752
        I = $UNFMI(I,2,SEED)                                          00007780    778                 753
        IF(I .GT. 1) GO TO 100                                        00007790    779                 754
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(OBTAN,DMY,MOSC,CALC); 00007800    780                 755
        WAIT RETURN: SERVICE=INTF;                                    00007810    781                 756
  100 CONTINUE                                                        00007820    782                 757
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(STOR,DMY,IDEN,DMY);   00007830    783                 758
        WAIT RETURN: SERVICE=INTF;                                    00007840    784                 759
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(STOR,DMY,ADDI,DMY);   00007850    785                 760
        WAIT RETURN: SERVICE=INTF;                                    00007860    786                 761
C       DETERMINE IF COMPATABILITY CHECK HAS OCCURRED                 00007870    787                 762
        RN = $UNFM(0.0, 1.0, SEED)                                    00007880    788                 763
        IF(RN .LT. PERR) GO TO 300                                    00007890    789                 764
C       NO COMPATABILITY ERROR                                        00007900    790                 765
        INVOKE: SERVICE= INTFS, PARAMETER LIST=(STOR,DMY,DIST,DMY);   00007910    791                 766
        WAIT RETURN: SERVICE=INTF;                                    00007920    792                 767

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                         INPUT CARD IMAGE                                            INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80    SEQ-NO   LEV SEQ-NO  REF-NO

       INVOKE: SERVICE= INTF$, PARAMETER LIST=(FINDR,NEXT,GRADE,UMGR);    00007930    793                 768
       WAIT RETURN: SERVICE=INTF;                                         00007940    794                 769
       INVOKE: SERVICE= INTF$, PARAMETER LIST=(CONCT,DMY,UIST,GRU1);      00007950    795                 770
       WAIT RETURN: SERVICE=INTF;                                         00007960    796                 771
       INVOKE: SERVICE= INTF$, PARAMETER LIST=(FINDR,DMY,MOSC,CALC);      00007970    797                 772
       WAIT RETURN: SERVICE=INTF;                                         00007980    798                 773
       INVOKE: SERVICE= INTF$, PARAMETER LIST=(CONCT,DMY,IDEN,PMID);      00007990    799                 774
       WAIT RETURN: SERVICE=INTF;                                         00008000    800                 775
       I = $UNFMI(1,2,SEED)                                               00008010    801                 776
       IF(I .GT. 1) GO TO 200                                             00008020    802                 777
       INVOKE: SERVICE= INTF$, PARAMETER LIST=(FINDR,DMY,MOSC,CALC);      00008030    803                 778
       WAIT RETURN: SERVICE=INTF;                                         00008040    804                 779
       INVOKE: SERVICE= INTF$, PARAMETER LIST=(CONCT,DMY,IDEN,SMID);      00008050    805                 780
       WAIT RETURN: SERVICE=INTF;                                         00008060    806                 781
   200 CONTINUE                                                           00008070    807                 782
C      NORMAL RETURN                                                      00008080    808                 783
       TLOC(6) = 0                                                        00008090    809                 784
       GO TO 350                                                          00008100    810                 785
C      COMPATABILITY ERROR                                                00008110    811                 786
   300 CONTINUE                                                           00008120    812                 787
       TLOC(6) = 1                                                        00008130    813                 788
   350 CONTINUE                                                           00008140    814                 789
       RELEASE: FACILITY=ADDSL;                                           00008150    815                 790
       END: ENDO SERVICE;                                                 00008160    816                 791

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                    PAGE  35
                               STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                        INPUT CARD IMAGE                                          INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

                                                                      00008170    817
                                                                      00008180    818
      ENDO SERVICE: ID=INTF, NAME= INTF$, SAVE AREA SIZE=10,          00008190    819                 792
                 PARAMETER LIST = (P1,P2,P3,P4);                      00008200    820                 793
C     IDMS INTERFACE ROUTINE                                          00008210    821                 794
      COMMON /WK/ TLOC                                                00008220    822                 795
      INTEGER ERRST,CIND,VALERR                                       00008230    823                 796
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4                             00008240    824                 797
      INTEGER $UNFM1,SEED,SEED1,TLOC(10)                              00008250    825                 798
      INTEGER OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS                 00008260    826                 799
      INTEGER DMY,NEXT,PRIOR,OWNER                                    00008270    827                 800
      INTEGER ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA      00008280    828                 801
      INTEGER CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI 00008290    829                 802
      EQUIVALENCE                           ($SAVE(5),DML),           00008300    830                 803
     1           ($SAVE(6),OPER), ($SAVE(7),RT),   ($SAVE(8),SET)     00008310    831                 804
      REAL $UNFM,RN                                                   00008320    832                 805
      DATA OBTAN,FINDR,MODIF,STOR,CONCT,DISCN,ERAS/1,2,3,4,5,6,7/     00008330    833                 806
      DATA DMY,NEXT,PRIOR,OWNER/0,1,2,3/                              00008340    834                 807
      DATA ADDI,ADUN,ALOC,GRADE,IDEN,INST,MOSC,POSN,UIST,UNMA         00008350    835                 808
     1 /   1,   2,   3,    4,   5,   6,   7,   8,   9,  10/           00008360    836                 809
      DATA CALC,GRUI,IDUI,INUM,PMID,POUI,SMID,UMAW,UMGR,UMPOS,UMUI    00008370    837                 810
     1 /   1,   2,   3,   4,   5,   6,   7,   8,   9,   10,  11/      00008380    838                 811
      END: DECLARATIONS;                                              00008390    839                 812
      DML  = P1                                                       00008400    840                 [illegible]

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                            INPUT CARD IMAGE                                           INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

        OPFR = P2                                                       00008410     841                  814
        RT   = P3                                                       00008420     842                  815
        SET  = P4                                                       00008430     843                  816
        END: INITIALIZATION;                                            00008440     844                  817
        SEIZE: FACILITY=INTF;                                           00008450     845                  818
C       STEP 1.  PERFORM FUNCTION VALIDITY CHECKING AND SEGMENTATION   00008460     846                  819
C         (SYSTEM HOOK)                                                 00008470     847                  820
          VALERR = 0                                                    00008480     848                  821
C       STEP 2.  TEST ERROR STATUS AFTER VALIDITY CHECKING              00008490     849                  822
            IF(VALERR .NE. 0) GO TO 80                                  00008500     850                  823
C           NO VALIDITY ERRORS OCCURRED                                 00008510     851                  824
C       STEP 3.  PERFORM ANY 'BEFORE' PROCEDURES                        00008520     852                  825
C         (SYSTEM HOOK)                                                 00008530     853                  826
C         NOTE - ERRST IS THE 'ERROR STATUS' INDICATOR                  00008540     854                  827
C                CIND  IS THE 'CANCEL' INDICATOR                        00008550     855                  828
          ERRSY = 0                                                     00008560     856                  829
          CIND = 0                                                      00008570     857                  830
C       STEP 4.  TEST THE ERROR STATUS FROM THE 'BEFORE' PROCEDURES     00008580     858                  831
C         (SYSTEM HOOK)                                                 00008590     859                  832
          IF(ERRST .NE. 0) GO TO 80                                     00008600     860                  833
C         NO ERRORS OCCURRED                                            00008610     861                  834
C       STEP 5.  TEST THE CANCEL INDICATOR                              00008620     862                  835
          IF(CIND .NE. 0) GO TO 100                                     00008630     863                  836
C         DML NOT CANCELLED                                             00008640     864                  837

........10........20........30........40........50........60........70........80
```

```
                    INFORMATION PROCESSING SYSTEM SIMULATOR                                      PAGE  27
                         STANDARD INPUT STREAM LISTING
                              DATE ... 09/10/77
                       INPUT CARD IMAGE                                       INPUT    ALT INPUT  MEMBER
.........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

C     STEP 6.  PERFORM DATA BASE MANAGEMENT FUNCTION                  00008650    865              838
C       ALL DML'S ARE ASSUMED TO REQUIRE DBMS SERVICE                 00008660    866              839
C       (SYSTEM HOOK FOR FRONT-END BACK-END COMMUNICATION)            00008670    867              840
        INVOKE: SERVICE=CAMPS, PARAMETER LIST=(DML,OPER,RT,SET);      00008680    868              841
        WAIT RETURN: SERVICE=CAMP;                                    00008690    869              842
C     STEP 7.  TEST ERROR STATUS                                      00008700    870              843
        IF(ERRST .EQ. 0) GO TO 100                                    00008710    871              844
C       ERROR OCCURRED IN THE EXECUTION OF THE DML                    00008720    872              845
C     STEP 8.  EXECUTE ANY 'ON ERROR DURING' PROCEDURES               00008730    873              846
C       (SYSTEM HOOK)                                                 00008740    874              847
   80   CONTINUE                                                      00008750    875              848
C     STEP 9.  EXECUTE ANY 'AFTER ERROR' PROCEURES                    00008760    876              849
C       (SYSTEM HOOK)                                                 00008770    877              850
C     STEP 10.  PERFORM SEGMENTATION                                  00008780    878              851
C       (SYSTEM HOOK)                                                 00008790    879              852
  100   CONTINUE                                                      00008800    880              853
C     STEP 11.  RETURN CONTROL TO THE APPLICATION PROGRAM             00008810    881              854
        RELEASE: FACILITY=INTF;                                       00008820    882              855
      END: ENDO SERVICE;                                              00008830    883              856
                                                                      00008840    884
                                                                      00008850    885
      ENDO SERVICE: ID=CAMP, NAME= CAMPS, SAVE AREA SIZE=10,          00008860    886              857
                    PARAMETER LIST = (P1,P2,P3,P4);                   00008870    887              858
C       ********* CENTRAL ACCESS MONITOR PROGRAM *********            00008880    888              859

.........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                    PAGE  26
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                  INPUT CARD IMAGE                                           INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

C      THIS PROGRAM 'THREADS REQUESTS AS (THE) DBMS CAN ACCOMODATE   00008890     889                 860
C      THEM - HOLDING THE NECESSARY USERS IN THE WAIT STATE'          00008900     890                 861
C                                                                     00008910     891                 862
      COMMON /WK/ TLOC                                                00008920     892                 863
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10)                    00008930     893                 864
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),     00008940     894                 865
     1            ($SAVE(4),SET)                                      00008950     895                 866
      END: DECLARATIONS;                                              00008960     896                 867
      DML  = P1                                                       00008970     897                 868
      OPER = P2                                                       00008980     898                 869
      RT   = P3                                                       00008990     899                 870
      SET  = P4                                                       00009000     900                 871
      END: INITIALIZATION;                                            00009010     901                 872
      SEIZE: FACILITY=CAMP;                                           00009020     902                 873
        QUEUE: FACILITY=DBMS;                                         00009030     903                 874
        WAIT FACILITY: FACILITY=DBMS, STATUS=0, CONDITION=EQ;         00009040     904                 875
        DEPART QUEUE:  FACILITY=DBMS;                                 00009050     905                 876
        SET STATUS: FACILITY=DBMS, STATUS=1;                          00009060     906                 877
          INVOKE: SERVICE=DBMS$, PARAMETER LIST=(DML,OPER,RT,SET);    00009070     907                 878
          WAIT RETURN: SERVICE=DBMS;                                  00009080     908                 879
        SET STATUS: FACILITY=DBMS, STATUS=0;                          00009090     909                 880
      RELEASE: FACILITY=CAMP;                                         00009100     910                 881
      END: ENDJ SERVICE;                                              00009110     911                 882
                                                                      00009120     912

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                      PAGE   39
                             STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                       INPUT CARD IMAGE                                        INPUT    ALT INPUT   MEMBER
.........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

                                                                      00009130      913
      ENDD SERVICE: ID=DBMS, NAME= DBMS$, SAVE AREA SIZE=10,          00009140      914                883
                      PARAMETER LIST = (P1,P2,P3,P4);                 00009150      915                884
C     ******** DATA BASE MANAGEMENT SYSTEM FUNCTIONS********          00009160      916                885
      COMMON /WK/ TLOC                                                00009170      917                886
      COMMON /PE/ PAGFT                                               00009180      918                887
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG 00009190    919                888
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                00009200      920                889
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX               00009210      921                890
      INTEGER I,J,RN,SUNFMI,RETP,TOT1,TOT2,PAGFT(10)                  00009220      922                891
      EQUIVALENCE ($SAVE(5),DML), ($SAVE(6),OPER), ($SAVE(7),RT),     00009230      923                892
     1                ($SAVE(8),SET)                                  00009240      924                893
C     BUFFER - BUF(10,3) - FIRST SUBSCRIPT IS BUFFER NUMBER           00009250      925                894
C             - SECOND SUBSCRIPT                                      00009260      926                895
C                     (1) PAGE NUMBER (0 TO NUMBER OF PAGES IN DB)    00009270      927                896
C                     (2) LEAST RECENTLY USED (LRU) INDICATOR (0-NPAG) 00009280     928                897
C                         LRU PAGE HAS VALUE 0, NEWEST PAGE = NPAG    00009290      929                898
C                     (3) MUST REWRITE INDICATOR (0-1)                00009300      930                899
C                         REWRITE PAGE WHEN = 1                       00009310      931                900
C     PAGE FAULT STATISTICS AREA - PAGFT(10) -                        00009320      932                901
C      (1) NUMBER OF PAGES CONCURRENTLY IN THE PAGE BUFFER            00009330      933                902
C      (2) PAGE SIZE (BYTES)                                          00009340      934                903
C      (3) PAGE WRITES                                                00009350      935                904
C      (4) CACHE MISSES ON WRITE                                      00009360      936                905

.........10........20........30........40........50........60........70........80
```

```
                              INFORMATION PROCESSING SYSTEM SIMULATOR                                      PAGE  40
                                   STANDARD INPUT STREAM LISTING
                                       DATE ... 09/10/77
                         INPUT CARD IMAGE                                    INPUT    ALT INPUT  MERGE
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO REF-NO

C       (5) PAGE READS                                                    00009370   937              906
C       (6) CACHE MISSES ON READ                                          00009380   938              907
C       (7) NUMBER OF PAGES REWRITEN BEFORE PAGE SPACE REUSED             00009390   939              908
C       (8) NUMBER OF PAGE REPLACEMENTS WITHOUT REWRITTING                00009400   940              909
C       (9) NUMBER OF PAGE NUMBERS GENERATED (BY GENPR)                   00009410   941              910
C       (10) TOTAL DISTANCE BETWEEN PAGE REFERENCES                       00009420   942              911
      END: DECLARATIONS;                                                  00009430   943              912
      DML  = P1                                                           00009440   944              913
      OPER = P2                                                           00009450   945              914
      RT   = P3                                                           00009460   946              915
      SET  = P4                                                           00009470   947              916
      IF(PAGFT(2) .EQ. 1024) GO TO 18                                     00009480   948              917
      DO 15 I = 1,10                                                      00009490   949              918
      DO 10 J = 1,3                                                       00009500   950              919
   10 BUF(I,J) = 0                                                        00009510   951              920
      DO 15 I = 1,10                                                      00009520   952              921
   15 PAGFT(I) = 0                                                        00009530   953              922
C     PAGE SIZE IS 1024 BYTES ON THE PDP11/70 AND 3156 ON THE IBM370/16500009540   954              923
      PAGFT(2) = 1024                                                     00009550   955              924
      FSTPN = 1000                                                        00009560   956              925
      LSTPN = 1200                                                        00009570   957              926
      AVEDIS = 45                                                         00009580   958              927
      RNRET = 1000                                                        00009590   959              928
      NPAG = 5                                                            00009600   960              929

........10........20........30........40........50........60........70........80
```

```
                        INFORMATION PROCESSING SYSTEM SIMULATOR                               PAGE   41

                             STANDARD INPUT STREAM LISTING

                                  DATE ... 09/10/77

                   INPUT CARD IMAGE                                              INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

      PAGFT(I) = NPAG                                                   00009610   961                 930
      INDX = 0                                                          00009620   962                 931
   18 CONTINUE                                                          00009630   963                 932
      END: INITIALIZATION;                                              00009640   964                 933
      SEIZE: FACILITY=DBMS;                                             00009650   965                 934
C     INVOKE THE REQUESTED DML SERVICE                                  00009660   966                 935
      GO TO (100,200,300,400,500,600,700),DML                           00009670   967                 936
C     OBTAIN DML                                                        00009680   968                 937
  100 CONTINUE                                                          00009690   969                 938
      INVOKE: SERVICE=OBTAIS, PARAMETER LIST=(DML,OPER,RT,SET);         00009700   970                 939
      WAIT RETURN: SERVICE=OBTAIN;                                      00009710   971                 940
      GO TO 900                                                         00009720   972                 941
C     FIND DML                                                          00009730   973                 942
  200 CONTINUE                                                          00009740   974                 943
      INVOKE: SERVICE=FINDS, PARAMETER LIST=(DML,OPER,RT,SET);          00009750   975                 944
      WAIT RETURN: SERVICE=FIND;                                        00009760   976                 945
      GO TO 900                                                         00009770   977                 946
C     MODIFY DML                                                        00009780   978                 947
  300 CONTINUE                                                          00009790   979                 948
      INVOKE: SERVICE=MODIFS, PARAMETER LIST=(DML,OPER,RT,SET);         00009800   980                 949
      WAIT RETURN: SERVICE=MODIFY;                                      00009810   981                 950
      GO TO 900                                                         00009820   982                 951
C     STORE DML                                                         00009830   983                 952
  400 CONTINUE                                                          00009840   984                 953

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                              PAGE  [illegible]
                              STANDARD INPUT STREAM LISTING
                                    DATE ... 09/10/77
                        INPUT CARD IMAGE                                     INPUT    ALT INPUT  MERGER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

      INVOKE: SERVICE=STORE$, PARAMETER LIST=(DML,OPER,RT,SET);         00009850    985              954
      WAIT RETURN: SERVICE=STORE;                                       00009860    986              955
      GO TO 900                                                         00009870    987              956
C     CONNECT DML                                                       00009880    988              957
  500 CONTINUE                                                          00009890    989              958
      INVOKE: SERVICE=CONEC$, PARAMETER LIST=(DML,OPER,RT,SET);         00009900    990              959
      WAIT RETURN: SERVICE=CONECT;                                      00009910    991              960
      GO TO 900                                                         00009920    992              961
C     DISCONNECT DML                                                    00009930    993              962
  600 CONTINUE                                                          00009940    994              963
      INVOKE: SERVICE=DISCO$, PARAMETER LIST=(DML,OPER,RT,SET);         00009950    995              964
      WAIT RETURN: SERVICE=DISCON;                                      00009960    996              965
      GO TO 900                                                         00009970    997              966
C     ERASE DML                                                         00009980    998              967
  700 CONTINUE                                                          00009990    999              968
      INVOKE: SERVICE=ERASE$, PARAMETER LIST=(DML,OPER,RT,SET);         00010000   1000              969
      WAIT RETURN: SERVICE=ERASE;                                       00010010   1001              970
C                                                                       00010020   1002              971
  950 CONTINUE                                                          00010030   1003              972
      RELEASE: FACILITY=DBMS;                                           00010040   1004              973
      END: ENDO SERVICE;                                                00010050   1005              974
                                                                        00010060   1006
                                                                        00010070   1007
      ENDO SERVICE: ID=OBTAIN, NAME= OBTAYS, SAVE AREA SIZE=10,         00010080   1008              975

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                           PAGE   43
                              STANDARD INPUT STREAM LISTING
                                  DATE ... 09/10/77

                     INPUT CARD IMAGE                                         INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

               PARAMETER LIST = (P1,P2,P3,P4);                               00010090  1009              976
C     ******** DBMS OBTAIN DML PROCESSING FUNCTIONS********                  00010100  1010              977
      COMMON /WK/ TLOC                                                       00010110  1011              978
      COMMON /PE/ PAGFT                                                      00010120  1012              979
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,POREC,BUF,RNRET,INDX,NPAG       00010130  1013              980
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                       00010140  1014              981
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX                      00010150  1015              982
      INTEGER I,J,RN,$UNFM1,RETP,TOT1,TOT2,PAGFT(10)                         00010160  1016              983
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),            00010170  1017              984
     I            ($SAVE(4),SET)                                             00010180  1018              985
      END: DECLARATIONS;                                                     00010190  1019              986
      DML  = P1                                                              00010200  1020              987
      OPER = P2                                                              00010210  1021              988
      RT   = P3                                                              00010220  1022              989
      SET  = P4                                                              00010230  1023              990
      END: INITIALIZATION;                                                   00010240  1024              991
      SEIZE: FACILITY=OBTAIN;                                                00010250  1025              992
C                                                                            00010260  1026              993
C     GENERATE A RANDOM PAGE NUMBER 1001-1200 - RETURNED VALUE IS RNRET      00010270  1027              994
   20 CALL GENRP                                                             00010280  1028              995
C     GET THIS PAGE IF NOT ALREADY PRESENT IN THE CACHE                      00010290  1029              996
      INVOKE: SERVICE=PAGMR$, PARAMETER LIST=(1);                            00010300  1030              997
      WAIT RETURN: SERVICE=PAGMR;                                            00010310  1031              998
C     DETERMINE IF THE NEXT PAGE IN THE CALC SET NEEDS TO BE ACCESSED        00010320  1032              999

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                                INPUT CARD IMAGE                                      INPUT   ALT INPUT   MEMBER
.........10........20........30........40........50........60........70........80     SEQ-NO  LEV SEQ-NO  REF-NO

C        THIS IS THE CASE ON 'FIND' AND/OR 'ACQUIRE' FOR CALC SETS        C0010330   1033                1000
C        WHEN THE RECORD COULD NOT BE PLACED ON THE PAGE DETERMINED BY    00010340   1034                1001
C        THE CALC ALGORITHM                                               00010350   1035                1002
      RN = $UNFMI(1,1000,2)                                               00010360   1036                1003
      IF(RN .GT. 900) GO TO 20                                            00010370   1037                1004
C     NEXT PAGE IN THE CALC SET NEED NOT BE ACCESSED                      00010380   1038                1005
C     PROCESSING OF 'OBTAIN' FINISHED                                     00010390   1039                1006
      RELEASE: FACILITY=OBTAIN;                                           00010400   1040                1007
      END: ENDO SERVICE;                                                  00010410   1041                1008
                                                                          00010420   1042
                                                                          00010430   1043
      ENDO SERVICE: ID=FIND, NAME= FINDS, SAVE AREA SIZE=10,              00010440   1044                1009
                    PARAMETER LIST = (P1,P2,P3,P4);                       00010450   1045                1010
C     ******** DBMS OBTAIN DML PROCESSING FUNCTIONS********               00010460   1046                1011
      COMMON /WK/ TLOC                                                    00010470   1047                1012
      COMMON /PE/ PAGFT                                                   00010480   1048                1013
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG  00010490   1049                1014
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                    00010500   1050                1015
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX                   00010510   1051                1016
      INTEGER I,J,RN,$UNFMI,RETP,TOT1,TOT2,PAGFT(10)                      00010520   1052                1017
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),         00010530   1053                1018
     I            ($SAVE(4),SET)                                          00010540   1054                1019
      END: DECLARATIONS;                                                  00010550   1055                1020
      DML  = P1                                                           00010560   1056                1021

.........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                      PAGE   45
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                    INPUT CARD IMAGE                                     INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  FIL-NO

      OPER = P2                                                  00010570   1057                1022
      RT   = P3                                                  00010580   1058                1023
      SET  = P4                                                  00010590   1059                1024
      END: INITIALIZATION;                                       00010600   1060                1025
      SEIZE: FACILITY=FIND;                                      00010610   1061                1026
C     GENERATE A RANDOM PAGE NUMBER 1001-1200                    00010620   1062                1027
   20 CALL GENRP                                                 00010630   1063                1028
C     GET THIS PAGE IF NOT ALREADY PRESENT IN THE CACHE          00010640   1064                1029
      INVOKE: SERVICE=PAGMR$, PARAMETER LIST=(I);                00010650   1065                1030
      WAIT RETURN: SERVICE=PAGMR;                                00010660   1066                1031
C     DETERMINE IF THE NEXT PAGE IN THE CALC SET NEEDS TO BE ACCESSED 00010670 1067             1032
C       THIS IS THE CASE ON 'FIND' AND/OR 'OBTAIN' FOR CALC SETS 00010680   1068                1033
C       WHEN THE RECORD COULD NOT BE PLACED ON THE PAGE DETERMINED BY 00010690 1069             1034
C       THE CALC ALGORITHM                                       00010700   1070                1035
      RN = $UNFMI(1,1000,3)                                      00010710   1071                1036
      IF(RN .GT. 900) GO TO 20                                   00010720   1072                1037
C     NEXT PAGE IN THE CALC SET NEED NOT BE ACCESSED             00010730   1073                1038
C     PROCESSING OF 'FIND' FINISHED                              00010740   1074                1039
  500 CONTINUE                                                   00010750   1075                1040
      RELEASE: FACILITY=FIND;                                    00010760   1076                1041
      END: ENDO SERVICE;                                         00010770   1077                1042
                                                                 00010780   1078
                                                                 00010790   1079
      ENDO SERVICE: ID=MODIFY, NAME= MODIF$, SAVE AREA SIZE=10,  00010800   1080                1043
........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                          PAGE   46
                               STANDARD INPUT STREAM LISTING
                                    DATE ... 09/10/77
                                INPUT CARD IMAGE                                  INPUT   ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

                  PARAMETER LIST = (P1,P2,P3,P4);                       00010810   1081              1044
C     ******** DBMS MODIFY DML PROCESSING FUNCTIONS********             00010820   1082              1045
      COMMON /WK/ TLOC                                                  00010830   1083              1046
      COMMON /PE/ PAGFT                                                 00010840   1084              1047
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG 00010850   1085              1048
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                  00010860   1086              1049
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX                 00010870   1087              1050
      INTEGER I,J,RN,$UNFMI,RETP,TOT1,TOT2,PAGFT(10)                    00010880   1088              1051
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),       00010890   1089              1052
     I            ($SAVE(4),SET)                                        00010900   1090              1053
      END: DECLARATIONS;                                                00010910   1091              1054
      DML  = P1                                                         00010920   1092              1055
      OPER = P2                                                         00010930   1093              1056
      RT   = P3                                                         00010940   1094              1057
      SET  = P4                                                         00010950   1095              1058
      END: INITIALIZATION;                                              00010960   1096              1059
      SEIZE: FACILITY=MODIFY;                                           00010970   1097              1060
C     WALGREEN DATA - 50% READ & 100% WRITE MISSES                      00010980   1098              1061
C     (SYSTEM HOOK)                                                     00010990   1099              1062
      PROCESS: TIME=14.7;                                               00011000   1100              1063
      RELEASE: FACILITY=MODIFY;                                         00011010   1101              1064
      END: ENDO SERVICE;                                                00011020   1102              1065
                                                                        00011030   1103
      ENDO SERVICE: ID=STORE, NAME= STORES, SAVE AREA SIZE=10,          00011040   1104              1066

........10........20........30........40........50........60........70........80
```

```
                              INPUT CARD IMAGE                                          INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO   REF-NO

                    PARAMETER LIST = (P1,P2,P3,P4);                          00011050    1105               1067
C     ******** DBMS STORE DML PROCESSING FUNCTIONS********                   00011060    1106               1068
      COMMON /WK/ TLOC                                                       00011070    1107               1069
      COMMON /PE/ PAGFT                                                      00011080    1108               1070
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG     00011090    1109               1071
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                       00011100    1110               1072
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX                      00011110    1111               1073
      INTEGER I,J,RN,$UNFMI,RETP,TOT1,TOT2,PAGFT(10)                         00011120    1112               1074
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),           00011130    1113               1075
     I           ($SAVE(4),SET)                                              00011140    1114               1076
      END: DECLARATIONS;                                                     00011150    1115               1077
      DML  = P1                                                              00011160    1116               1078
      OPER = P2                                                              00011170    1117               1079
      RT   = P3                                                              00011180    1118               1080
      SET  = P4                                                              00011190    1119               1081
      END: INITIALIZATION;                                                   00011200    1120               1082
      SEIZE: FACILITY=STORE;                                                 00011210    1121               1083
C     DETERMINE IF THE 'SPACE MANAGEMENT PAGE' NEEDS TO BE ACCESSED          00011220    1122               1084
C     (ON 'STORE' WHEN NOT ENOUGH SPACE AVAILABLE IN PRESENT PAGE)           00011230    1123               1085
      RN = $UNFMI(1,1000,4)                                                  00011240    1124               1086
      IF(RN .LT. 990) GO TO 200                                              00011250    1125               1087
C     MUST ACCESS THE SPACE MANAGEMENT PAGE                                  00011260    1126               1088
C     GET THIS PAGE IF NOT ALREADY IN THE CACHE                              00011270    1127               1089
      RNRET = FSTPN                                                          00011280    1128               1090

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77


```
                          INPUT CARD IMAGE                             INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

         INVOKE: SERVICE=PAGMRS, PARAMETER LIST=(1);               00011290   1129          1091
         WAIT RETURN: SERVICE=PAGMR;                               00011300   1130          1092
C        ACCESS THE PAGE ON WHICH SPACE IS AVALIABLE(POINTED TO BY THE 00011310 1131        1093
C        SPACE MANAGEMENT PAGE)                                    00011320   1132          1094
C        GENERATE A RANDOM PAGE NUMBER                             00011330   1133          1095
         CALL GENRP                                                00011340   1134          1096
C        GET SPACE-AVAILABLE PAGE IF NOT ALREADY IN THE CACHE      00011350   1135          1097
         INVOKE: SERVICE=PAGMRS, PARAMETER LIST=(2);               00011360   1136          1098
         WAIT RETURN: SERVICE=PAGMR;                               00011370   1137          1099
C        DETERMINE IF THE 'SPACE MANAGEMENT PAGE' MUST BE REWITTEN 00011380   1138          1100
C        (ONLY WHEN PAGE IS > 70% FULL)                            00011390   1139          1101
         RN = $UNFMI(1,1000,4)                                     00011400   1140          1102
         IF(RN .LT. 990) GO TO 200                                 00011410   1141          1103
C        REWITE NO-SPACE-AVALIABLE PAGE TO UPDATE POINTER TO NEXT IN CALC 00011420 1142     1104
C        SET                                                       00011430   1143          1105
C        DETERMINE IF NO-SPACE-AVAILABLE PAGE IS IN THE CACHE      00011440   1144          1106
  200    CONTINUE                                                  00011450   1145          1107
         INVOKE: SERVICE=PAGMRS, PARAMETER LIST=(2);               00011460   1146          1108
         WAIT RETURN: SERVICE=PAGMR;                               00011470   1147          1109
         RELEASE: FACILITY=STORE;                                  00011480   1148          1110
         END: ENDO SERVICE;                                        00011490   1149          1111
                                                                   00011500   1150
                                                                   00011510   1151
         ENDO SERVICE: ID=CONECT, NAME= CONEC$, SAVE AREA SIZE=10, 00011520   1152          1112

........10........20........30........40........50........60........70........80
```

THIS PAGE IS BEST QUALITY PRACTICABLE
FROM COPY FURNISHED TO DDC

```
                      INFORMATION PROCESSING SYSTEM SIMULATOR                         PAGE  49
                          STANDARD INPUT STREAM LISTING
                               DATE ... 09/10/77
                      INPUT CARD IMAGE                                 INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

                 PARAMETER LIST = (P1,P2,P3,P4);                      00011530   1153               1113
C      ******** DBMS CONNECT DML PROCESSING FUNCTIONS********        00011540   1154               1114
       COMMON /WK/ TLOC                                               00011550   1155               1115
       COMMON /PE/ PAGFT                                              00011560   1156               1116
       COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG 00011570   1157            1117
       INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)               00011580   1158               1118
       INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX              00011590   1159               1119
       INTEGER I,J,RN,$UNFM1,RETP,TOT1,TOT2,PAGFT(10)                 00011600   1160               1120
       EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),   00011610   1161               1121
      1           ($SAVE(4),SET), ($SAVE(7),RN), ($SAVE(8),I)        00011620   1162               1122
       END: DECLARATIONS;                                             00011630   1163               1123
       DML  = P1                                                      00011640   1164               1124
       OPER = P2                                                      00011650   1165               1125
       RT   = P3                                                      00011660   1166               1126
       SET  = P4                                                      00011670   1167               1127
       END: INITIALIZATION;                                           00011680   1168               1128
       SEIZE: FACILITY=CONECT;                                        00011690   1169               1129
C      ASSUME UNIFORM DISTRIBUTION OF 1-5 MEMBER RECORDS TO BE CONNECTED 00011700   1170            1130
       RN = $UNFM1(1,5,5)                                             00011710   1171               1131
       I = 0                                                          00011720   1172               1132
  150  I = I + 1                                                      00011730   1173               1133
       IF (I .GT. RN) GO TO 500                                       00011740   1174               1134
       CALL GENRP                                                     00011750   1175               1135
       INVOKE: SERVICE=PAGMRS, PARAMETER LIST=(I);                    00011760   1176               1136

........10........20........30........40........50........60........70........80
```

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                     INPUT CARD IMAGE                                          INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80       SEQ-NO  LEV SEQ-NO  REF-NO

      WAIT RETURN: SERVICE=FAGMR;                                    00011770  1177                1137
      GO TO 150                                                      00011780  1178                1138
  500 CONTINUE                                                       00011790  1179                1139
      RELEASE: FACILITY=CONECT;                                      00011800  1180                1140
      END: ENDO SERVICE;                                             00011810  1181                1141
                                                                     00011820  1182
                                                                     00011830  1183
      ENDO SERVICE: ID=DISCON, NAME= DYSCO$, SAVE AREA SIZE=10,      00011840  1184                1142
                   PARAMETER LIST = (P1,P2,P3,P4);                   00011850  1185                1143
C     ******** DBMS DISCONECT PROCESSING FUNCTIONS********           00011860  1186                1144
      COMMON /WK/ TLOC                                               00011870  1187                1145
      COMMON /PE/ PAGFT                                              00011880  1188                1146
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG 00011890  1189             1147
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)               00011900  1190                1148
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX              00011910  1191                1149
      INTEGER I,J,RN,$UNFMI,RETP,TOT1,TOT2,PAGFT(10)                 00011920  1192                1150
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),    00011930  1193                1151
     1            ($SAVE(4),SET), ($SAVE(7),RN), ($SAVE(6),I)        00011940  1194                1152
      END: DECLARATIONS;                                             00011950  1195                1153
      DML  = P1                                                      00011960  1196                1154
      OPER = P2                                                      00011970  1197                1155
      RT   = P3                                                      00011980  1198                1156
      SET  = P4                                                      00011990  1199                1157
      END: INITIALIZATION;                                           00012000  1200                1158

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                                PAGE   =1

                              STANDARD INPUT STREAM LISTING

                                   DATE ... 09/10/77

                             INPUT CARD IMAGE                                            INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80        SEQ-NO   LEV SEQ-NO REF-NO

      SEIZE: FACILITY=DISCON;                                               00012010    1201                 1159
C     ASSUME UNIFORM DISTRIBUTION OF 1-5 RECORDS TO BE DISCONNECTED         00012020    1202                 1160
      RN = $UNFMI(1,5,6)                                                    00012030    1203                 1161
      I = 0                                                                 00012040    1204                 1162
  150 I = I + 1                                                             00012050    1205                 1163
      IF (I .GT. RN) GO TO 500                                              00012060    1206                 1164
      CALL GENRP                                                            00012070    1207                 1165
      INVOKE: SERVICE=PAGMR$, PARAMETER LIST=(1);                           00012080    1208                 1166
      WAIT RETURN: SERVICE=PAGMR;                                           00012090    1209                 1167
      GO TO 150                                                             00012100    1210                 1168
  500 CONTINUE                                                              00012110    1211                 1169
      RELEASE: FACILITY=DISCON;                                             00012120    1212                 1170
      END: ENDO SERVICE;                                                    00012130    1213                 1171
                                                                            00012140    1214
                                                                            00012150    1215
      ENDO SERVICE: ID=ERASE, NAME= ERASE$, SAVE AREA SIZE=10,              00012160    1216                 1172
                     PARAMETER LIST = (P1,P2,P3,P4);                        00012170    1217                 1173
C     ******** DBMS ERASE DML PROCESSING FUNCTIONS********                  00012180    1218                 1174
      COMMON /WK/ TLOC                                                      00012190    1219                 1175
      COMMON /PE/ PAGFT                                                     00012200    1220                 1176
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG     00012210    1221                 1177
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                      00012220    1222                 1178
      INTEGER DML,OPER,KT,SET,P1,P2,P3,P4,TLOC(10),INDX                     00012230    1223                 1179
      INTEGER I,J,RN,$UNFMI,RETP,TOT1,TOT2,PAGFT(10)                        00012240    1224                 1180

........10........20........30........40........50........60........70........80
```

```
                              INFORMATION PROCESSING SYSTEM SIMULATOR                                PAGE  52

                                   STANDARD INPUT STREAM LISTING

                                          DATE ... 09/10/77

                               INPUT CARD IMAGE                                       INPUT   ALT INPUT   MEMBER
.........10........20........30........40........50........60........70........80    SEQ-NO  LEV SEQ-NO  REF-NO

      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),      00012250    1225              1181
     I            ($SAVE(4),SET), ($SAVE(5),RN), ($SAVE(6),I)          00012260    1226              1182
      END: DECLARATIONS;                                               00012270    1227              1183
      DML  = P1                                                        00012280    1228              1184
      OPER = P2                                                        00012290    1229              1185
      RT   = P3                                                        00012300    1230              1186
      SET  = P4                                                        00012310    1231              1187
      END: INITIALIZATION;                                             00012320    1232              1188
      SEIZE: FACILITY=ERASE;                                           00012330    1233              1189
C     WALGREEN DATA - 50% READ & 100% WRITE MISS RATIO                 00012340    1234              1190
C       'ERASE ALL MEMBERS' - .3%        (101,000 MSEC AVE)            00012350    1235              1191
C       'ERASE <RECORD NAME>' - 99.7%    (49 MSEC AVE)                 00012360    1236              1192
      RN = $UNFMI(1,1000,7)                                            00012370    1237              1193
      IF(RN .GT. 997) GO TO 100                                        00012380    1238              1194
C     PROCESS 'ERASE <RECORD NAME> DML                                 00012390    1239              1195
      CALL GENRP                                                       00012400    1240              1196
      INVOKE: SERVICE=PAGMR$, PARAMETER LIST=(I);                      00012410    1241              1197
      WAIT RETURN: SERVICE=PAGMR;                                      00012420    1242              1198
      GO TO 500                                                        00012430    1243              1199
C     PROCESS 'ERASE ALL MEMBERS' STATEMENT                            00012440    1244              1200
  100 CONTINUE                                                         00012450    1245              1201
C     ASSUME UNIFORM DISTRIBUTION OF 1 TO 5 MEMBER RECORDS             00012460    1246              1202
      RN = $UNFMI(1,5,7)                                               00012470    1247              1203
      I = 0                                                            00012480    1248              1204

.........10........20........30........40........50........60........70........80
```

```
                          INFORMATION PROCESSING SYSTEM SIMULATOR                                         PAGE   55

                              STANDARD INPUT STREAM LISTING

                                    DATE ... 09/10/77

                               INPUT CARD IMAGE                                          INPUT    ALT INPUT   MEMBER
 ........10........20........30........40........50........60........70........80     SEQ-NO   LEV SEQ-NO  REF-NO

     150 I = I + 1                                                            00012490   1249                  1205
         IF (I .GT. RN) GO TO 500                                             00012500   1250                  1206
         CALL GENRP                                                           00012510   1251                  1207
         INVOKE: SERVICE=PAGMRS, PARAMETER LIST=(I);                          00012520   1252                  1208
         WAIT RETURN: SERVICE=PAGMR;                                          00012530   1253                  1209
         GO TO 150                                                            00012540   1254                  1210
     500 CONTINUE                                                             00012550   1255                  1211
         RELEASE: FACILITY=ERASE;                                             00012560   1256                  1212
         END: ENDO SERVICE;                                                   00012570   1257                  1213
                                                                              00012580   1258
                                                                              00012590   1259
         ENDO SERVICE: ID=PAGMR, NAME= PAGMRS, PARAMETER LIST=(RW),           00012600   1260                  1214
                      SAVE AREA SIZE=10;                                      00012610   1261                  1215
 C        ****** BUFFER MANAGEMENT ROUTINE******                              00012620   1262                  1216
 C       (IDMS' MODIFIED LEAST-RECENTLY-USED ALGORITHM)                       00012630   1263                  1217
 C       THIS ROUTINE IS RESPONSIBLE FOR BRINGING THE REQUESTED PAGE          00012640   1264                  1218
 C       (I.E., RNRET) INTO THE PAGE BUFFER. THIS MAY INVOLVE REWRITING       00012650   1265                  1219
 C       ONE OF THE PAGES IN THE BUFFER.                                      00012660   1266                  1220
 C       INPUT PARAMETER - 1=READ, 2=WRITE                                    00012670   1267                  1221
         COMMON /WK/ TLOC                                                     00012680   1268                  1222
         COMMON /PE/ PAGFT                                                    00012690   1269                  1223
         COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG 00012700   1270                  1224
         INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                     00012710   1271                  1225
         INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX,RW,RDWT            00012720   1272                  1226

 ........10........20........30........40........50........60........70........80
```

```
                        INFORMATION PROCESSING SYSTEM SIMULATOR                              PAGE   54
                             STANDARD INPUT STREAM LISTING
                                  DATE ... 09/10/77
                      INPUT CARD IMAGE                                     INPUT   ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

      INTEGER I,J,RN,SUNFMI,RETP,TOT1,TOT2,PAGFT(10),IX             00012730   1273           1227
      EQUIVALENCE ($SAVE(1),DML), ($SAVE(2),OPER), ($SAVE(3),RT),   00012740   1274           1228
     1 ($SAVE(4),SET), ($SAVE(5),RN), ($SAVE(9),IX), ($SAVE(10),RDWT)  00012750   1275           1229
C     PAGE FAULT STATISTICS AREA - PAGFT(10) -                      00012760   1276           1230
C      (1) NUMBER OF PAGES CONCURRENTLY IN THE PAGE BUFFER          00012770   1277           1231
C      (2) PAGE SIZE (BYTES)                                        00012780   1278           1232
C      (3) PAGE WRITES                                              00012790   1279           1233
C      (4) CACHE MISSES ON WRITE                                    00012800   1280           1234
C      (5) PAGE READS                                               00012810   1281           1235
C      (6) CACHE MISSES ON READ                                     00012820   1282           1236
C      (7) NUMBER OF PAGES REWRITEN BEFORE PAGE SPACE REUSED        00012830   1283           1237
C      (8) NUMBER OF PAGE REPLACEMENTS WITHOUT REWRITTING           00012840   1284           1238
C      (9) NUMBER OF PAGE NUMBERS GENERATED (BY GENPR)              00012850   1285           1239
C      (10) TOTAL DISTANCE BETWEEN PAGE REFERENCES                  00012860   1286           1240
      END: DECLARATIONS;                                            00012870   1287           1241
      RN = RNRET                                                    00012880   1288           1242
      RDWT = RW                                                     00012890   1289           1243
      END: INITIALIZATION;                                          00012900   1290           1244
      SEIZE: FACILITY= PAGMR;                                       00012910   1291           1245
      IF(RDWT .LT. 1) RDWT = 1                                      00012920   1292           1246
      IF(RDWT .GT. 2) RDWT = 2                                      00012930   1293           1247
C     UPDATE NUMBER OF READS/WRITES                                 00012940   1294           1248
      IF(RDWT .EQ. 1) PAGFT(5) = PAGFT(5) + 1                       00012950   1295           1249
      IF(RDWT .EQ. 2) PAGFT(3) = PAGFT(3) + 1                       00012960   1296           1250

........10........20........30........40........50........60........70........80
```

```
                                   INFORMATION PROCESSING SYSTEM SIMULATOR                              PAGE   55
                                       STANDARD INPUT STREAM LISTING
                                           DATE ... 09/10/77
                           INPUT CARD IMAGE                                             INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80     SEQ-NO  LEV SEQ-NO  REF-NO

C     DETERMINE IF PAGE IS IN BUFFER POOL                                     00012970    1297              1251
      DO 10 I = 1,NPAG                                                        00012980    1298              1252
        IX = I                                                                00012990    1299              1253
        IF(BUF(I,1) .EQ. RN) GO TO 70                                         00013000    1300              1254
   10 CONTINUE                                                                00013010    1301              1255
C     PAGE WAS NOT IN THE BUFFER POOL                                         00013020    1302              1256
      IF(RDWT .EQ. 1) PAGFT(6) = PAGFT(6) + 1                                 00013030    1303              1257
      IF(RDWT .EQ. 2) PAGFT(4) = PAGFT(4) + 1                                 00013040    1304              1258
C     SIMULATE RECORD RETRIEVAL FROM DATA BASE - ELAPSED TIME = 36 MS.        00013050    1305              1259
C     (SYSTEM HOOK)                                                           00013060    1306              1260
      PROCESS: TIME=36.0;                                                     00013070    1307              1261
C     STORE THE PAGE IN THE BUFFER POOL                                       00013080    1308              1262
        IX = 0                                                                00013090    1309              1263
   30   IX = IX + 1                                                           00013100    1310              1264
        IF(IX .GT. NPAG) GO TO 70                                             00013110    1311              1265
        IF(BUF(IX,2) .NE. 0) GO TO 30                                         00013120    1312              1266
C       DETERMINE IF THE PAGE MUST BE WRITEN BEFORE IT IS REPLACED            00013130    1313              1267
        IF(BUF(IX,3) .EQ. 0) GO TO 40                                         00013140    1314              1268
C       PAGE TO BE REPLACED HAS BEEN MODIFIED - REWRITE                       00013150    1315              1269
        PAGFT(7) = PAGFT(7) + 1                                               00013160    1316              1270
C       SIMULATE RECORD STORAGE TO DATA BASE - ELAPSED TIME = 36 MS.          00013170    1317              1271
C       (SYSTEM HOOK)                                                         00013180    1318              1272
        PROCESS: TIME=36.0;                                                   00013190    1319              1273
        GO TO 45                                                              00013200    1320              1274

........10........20........30........40........50........60........70........80
```

```
                    INFORMATION PROCESSING SYSTEM SIMULATOR                                   PAGE  56
                        STANDARD INPUT STREAM LISTING
                             DATE ... 09/10/77
                          INPUT CARD IMAGE                                  INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

     40   PAGFT(8) = PAGFT(8) + 1                                  00013210    1321                 1275
C         AGE ALL EXISTING PAGES IN THE BUFFER POOL                00013220    1322                 1276
     45   DO 50 J = 1,NPAG                                         00013230    1323                 1277
            IF(BUF(J,2) .GT. 0) BUF(J,2) = BUF(J,2) - 1            00013240    1324                 1278
     50   CONTINUE                                                 00013250    1325                 1279
C         STORE THE NEW PAGE IN THE BUFFER POOL                    00013260    1326                 1280
          BUF(IX,1) = RN                                           00013270    1327                 1281
          BUF(IX,2) = NPAG - 1                                     00013280    1328                 1282
          BUF(IX,3) = 0                                            00013290    1329                 1283
C                                                                  00013300    1330                 1284
C                                                                  00013310    1331                 1285
C                                                                  00013320    1332                 1286
     70 INDX = IX                                                  00013330    1333                 1287
        IF(RDWT .EQ. 1) GO TO 80                                   00013340    1334                 1288
C       TURN MUST-WRITE SWITCH 'ON'                                00013350    1335                 1289
        BUF(INDX,3) = 1                                            00013360    1336                 1290
     80 CONTINUE                                                   00013370    1337                 1291
        RELEASE: FACILITY=PAGMR;                                   00013380    1338                 1292
        END: ENDO SERVICE;                                         00013390    1339                 1293
                                                                   00013400    1340
                                                                   00013410    1341
        PROCEDURE: NAME=GENRP, TYPE=SUBROUTINE;                    00013420    1342                 1294
C       THIS PROCEDURE GENERATES A UNIFORMLY DISTRIBUTED RANDOM PAGE 00013430  1343                 1295
C       NUMBER OVER THE SPACIFIED PAGE RANGE WITH THE DESIRED AVERAGE 00013440 1344                 1296

........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                      PAGE  [illegible]7
                               STANDARD INPUT STREAM LISTING
                                    DATE ... 09/10/77
                        INPUT CARD IMAGE                                        INPUT   ALT INPUT  MEMBER
.........10........20........30........40........50........60........70........80   SEQ-NO  LEV SEQ-NO  REF-NO

C     DISTANCE BETWEEN SUCCESSIVE PAGES.  USING THIS ALGORITHM,          00013450   1345               1297
C     'CLUSTERED' PAGE REFERENCES CAN BE GENERATED REGARDLESS OF THE     00013460   1346               1298
C     SIZE OF THE DATA BASE.                                             00013470   1347               1299
C                                                                        00013480   1348               1300
C     FSTPN - LOW ORDER BOUND ON GENERATED PAGE NUMBERS                  00013490   1349               1301
C     LSTPN - HIGH ORDER BOUND ON GENERATED PAGE NUMBERS                 00013500   1350               1302
C     AVEDIS - THE DESIRED AVERAGE DISTANCE BETWEEN SUCCESSIVE PAGE      00013510   1351               1303
C              REFERENCES                                                00013520   1352               1304
C     RNRET - THE COMPUTED PAGE NUMBER                                   00013530   1353               1305
      COMMON /WK/ TLOC                                                   00013540   1354               1306
      COMMON /PE/ PAGFT                                                  00013550   1355               1307
      COMMON /PAGAR/ FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF,RNRET,INDX,NPAG  00013560   1356               1308
      INTEGER FSTPN,LSTPN,AVEDIS,RNSAV,RDREC,BUF(10,3)                   00013570   1357               1309
      INTEGER DML,OPER,RT,SET,P1,P2,P3,P4,TLOC(10),INDX                  00013580   1358               1310
      INTEGER I,J,RN,$UNFMI,RETP,TOT1,TOT2,PAGFT(10),K,L                 00013590   1359               1311
      INTEGER RN1,RN2,VAR1,VAR2                                          00013600   1360               1312
      REAL $UNFM,Y,RN3                                                   00013610   1361               1313
C     INITIALIZE                                                         00013620   1362               1314
      MAXPN = LSTPN - FSTPN                                              00013630   1363               1315
      IF(MAXPN .LE. 0) GO TO 100                                         00013640   1364               1316
      RN1 = RNRET                                                        00013650   1365               1317
C     COMPUTE RN2 USING THE FULL RANGE OF PAGES                          00013660   1366               1318
   10 RN2 = $UNFMI(FSTPN,LSTPN,8)                                        00013670   1367               1319
      VAR1 = RN1 + AVEDIS - FSTPN                                        00013680   1368               1320

.........10........20........30........40........50........60........70........80
```

```
                              INPUT CARD IMAGE                                         INPUT    ALT INPUT    MEMBER
........10........20........30........40........50........60........70........80    SEQ-NO   LEV SEQ-NO   REF-NO

      IF(VAR1 .GT. MAXPN) VAR1 = MAXPN                                   00013690   1369                  1321
      VAR2 = RN1 - AVEDIS - FSTPN                                        00013700   1370                  1322
      IF(VAR2 .LT. 0) VAR2 = 0                                           00013710   1371                  1323
C     DON'T RECOMPUTE RN2 IF NOT OUTSIDE RANGE RN1+-AVEDIS               00013720   1372                  1324
      IF(RN2 .GT. RN1+AVEDIS) GO TO 20                                   00013730   1373                  1325
      IF(RN2 .LT. RN1-AVEDIS) GO TO 20                                   00013740   1374                  1326
      GO TO 50                                                           00013750   1375                  1327
   20 CONTINUE                                                           00013760   1376                  1328
C     Y = PROB(RN2 > RN1+VAR) OR PROB(RN2 < RN1-VAR)                     00013770   1377                  1329
      IF(RN2 .GT. RN1) Y = 1. - (VAR1/(MAXPN*1.0))                       00013780   1378                  1330
      IF(RN2 .LT. RN1) Y = (VAR2/(MAXPN*1.0))                            00013790   1379                  1331
C     TEST BOUNDS ON Y - ALGORITHM COULD LOOP - EXIT IF OUT OF RANGE     00013800   1380                  1332
      IF(Y .LT. 0.) GO TO 50                                             00013810   1381                  1333
      IF(Y .GT. 1.) GO TO 50                                             00013820   1382                  1334
C     GENERATE RN3 TO TEST AGAINST Y                                     00013830   1383                  1335
      RN3 = $UNFM(0.0,1.0,8)                                             00013840   1384                  1336
   30 CONTINUE                                                           00013850   1385                  1337
      IF(RN3 .GE. Y) GO TO 50                                            00013860   1386                  1338
C     RECOMPUTE RN2 - RANGE CRITERIA NOT MET (ON AVERAGE)                00013870   1387                  1339
      K = RN1 - AVEDIS                                                   00013880   1388                  1340
      IF(K .LT. FSTPN) K = FSTPN                                         00013890   1389                  1341
      L = RN1 + AVEDIS                                                   00013900   1390                  1342
      IF(L .GT. LSTPN) L = LSTPN                                         00013910   1391                  1343
      RN2 = $UNFMI(K,L,8)                                                00013920   1392                  1344

........10........20........30........40........50........60........70........80
```

```
                         STANDARD INPUT STREAM LISTING

                               DATE ... 09/10/77

                           INPUT CARD IMAGE                              INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80  SEQ-NO   LEV SEQ-NO REF-NO

      IDIF = (RN2-RN1)                                              00013930    1393              1345
C     AVERAGE DISTANCE CRITERIA IS SATISFIED - UPDATE VARIABLES & RETURN00013940 1394             1346
   50 RNRET = RN2                                                   00013950    1395              1347
      PAGFT(9) = PAGFT(9) + 1                                       00013960    1396              1348
      PAGFT(10) = PAGFT(10) + IABS(RN1-RN2)                         00013970    1397              1349
      IDIF = (RN2-RN1)                                              00013980    1398              1350
C     BYPASS SECOND WRITE OF GENRP DATA                             00013990    1399              1351
      IF(RN3 .LT. Y) GO TO 500                                      00014000    1400              1352
      GO TO 500                                                     00014010    1401              1353
  100 CONTINUE                                                      00014020    1402              1354
C     ERROR RETURN ON FSTPN LSTPN VALUES - FSTPN > LSTPN            00014030    1403              1355
      RNRET = $UNFMI(LSTPN,FSTPN,8)                                 00014040    1404              1356
  500 CONTINUE                                                      00014050    1405              1357
      RETURN                                                        00014060    1406              1358
      END: PROCEDURE;                                               00014070    1407              1359
                                                                    00014080    1408
                                                                    00014090    1409
      EXO SERVICE: ID=BUFST, NAME=BUFSTS,                           00014100    1410              1360
                        SAVE AREA SIZE = 10;                        00014110    1411              1361
C     PRINT BUFFER PAGE FAULT STATISTICS                            00014120    1412              1362
      COMMON /PE/ PAGFT                                             00014130    1413              1363
      INTEGER PAGFT(10),TOT1,TOT2                                   00014140    1414              1364
      REAL MISR                                                     00014150    1415              1365
      END: DECLARATIONS;                                            00014160    1416              1366

........10........20........30........40........50........60........70........80
```

AD-A069 543 OHIO STATE UNIV RESEARCH FOUNDATION COLUMBUS F/G 9/4

A SIMULATION LANGUAGE FOR EVALUATING INFORMATION PROCESSING SYS--ETC(U)

SEP 78 T G DELUTIS DAAG29-77-G-0203

UNCLASSIFIED ARO-15356.1-A-M NL

3 OF 3

AD A069543

END

DATE FILMED

7-79

DDC

STANDARD INPUT STREAM LISTING

DATE ... 09/10/77

```
                          INPUT CARD IMAGE                                         INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

      END: INITIALIZATION;                                                00014170   1417                1367
      WRITE(6,10)                                                         00014180   1418                1368
   10 FORMAT('1',20X,'CACHE PAGE FAULT STATISTICS'//)                     00014190   1419                1369
      WRITE(6,20) PAGFT(1)                                                00014200   1420                1370
   20 FORMAT(' ',10X,'CACHE SIZE (# PAGES):    ',I2)                      00014210   1421                1371
      WRITE(6,30) PAGFT(2)                                                00014220   1422                1372
   30 FORMAT(' ',10X,'PAGE SIZE (BYTES):     ',I4)                        00014230   1423                1373
      MISR = PAGFT(4)/(PAGFT(3)*1.0)                                      00014240   1424                1374
      WRITE(6,40) PAGFT(3),PAGFT(4),MISR                                  00014250   1425                1375
   40 FORMAT('0',10X,'PAGE WRITE REFERENCES: ',I6,                        00014260   1426                1376
     1       3X,'CACHE MISSES:',I6,3X,'MISS RATIO: ',F6.4)                00014270   1427                1377
      MISR = PAGFT(6)/(PAGFT(5)*1.0)                                      00014280   1428                1378
      WRITE(6,50) PAGFT(5),PAGFT(6),MISR                                  00014290   1429                1379
   50 FORMAT(' ',10X,'PAGE READ REFERENCES:  ',I6,                        00014300   1430                1380
     1       3X,'CACHE MISSES:',I6,3X,'MISS RATIO: ',F6.4)                00014310   1431                1381
      TOT1 = PAGFT(3) + PAGFT(5)                                          00014320   1432                1382
      TOT2 = PAGFT(4) + PAGFT(6)                                          00014330   1433                1383
      MISR = TOT2/(TOT1*1.0)                                              00014340   1434                1384
      WRITE(6,60) TOT1,TOT2,MISR                                          00014350   1435                1385
   60 FORMAT('0',10X,'PAGE TOTAL REFERENCES: ',I6,                        00014360   1436                1386
     1       3X,'CACHE MISSES:',I6,3X,'MISS RATIO: ',F6.4)                00014370   1437                1387
      WRITE(6,70) PAGFT(7)                                                00014380   1438                1388
   70 FORMAT('0',10X,'PAGES REWRITTEN BEFORE SPACE REUSED: ',I6)          00014390   1439                1389
      WRITE(6,80) PAGFT(8)                                                00014400   1440                1390

........10........20........30........40........50........60........70........80
```

```
                        INPUT CARD IMAGE                                      INPUT    ALT INPUT  MEMBER
........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO REF-NO

     80 FORMAT(' ',10X,'PAGES REPLACED WITHOUT REWRITING:   ',I6)   00014410   1441              1391
        MISR = PAGFT(10)/(PAGFT(9)*1.0)                             00014420   1442              1392
        WRITE(6,90) MISR                                            00014430   1443              1393
     90 FORMAT('0',10X,'AVERAGE DISTANCE BETWEEN SUCCESSIVE PAGE',  00014440   1444              1394
       1 ' REFERENCES: ',F9.4//)                                    00014450   1445              1395
        END: EXO SERVICE;                                           00014460   1446              1396
                                                                    00014470   1447
                                                                    00014480   1448
        EXO SERVICE: ID=SNAPST, NAME=SNAPS$,                        00014490   1449              1397
                      SAVE AREA SIZE = 10;                          00014500   1450              1398
C       PRINT STANDARD IPSS STATISTICS AT THE DESIGNATED INTERVALS  00014510   1451              1399
        END: DECLARATIONS;                                          00014520   1452              1400
        END: INITIALIZATION;                                        00014530   1453              1401
        CALL $SNAP                                                  00014540   1454              1402
        END: EXO SERVICE;                                           00014550   1455              1403
                                                                    00014560   1456
                                                                    00014570   1457
      END: SYSTEM RESOURCES;                                        00014580   1458              1404
                                                                    00014590   1459
                                                                    00014600   1460
      DEFINE EXOGENOUS EVENT STREAM: NAME=EX, COMPILE=YES,          00014610   1461
                            DISPOSITION=KEEP;                       00014620   1462
                                                                    00014630   1463
        EXOGENOUS EVENT: ID=EV1, TIME=PROC(TIME1);                  00014640   1464

........10........20........30........40........50........60........70........80
```

```
                           INFORMATION PROCESSING SYSTEM SIMULATOR                                  PAGE  62
                              STANDARD INPUT STREAM LISTING
                                   DATE ... 09/10/77
                      INPUT CARD IMAGE                                                   INPUT   ALT INPUT  REFER
........10........20........30........40........50........60........70........80      SEQ-NO  LEV SEQ-NO  REF-NO

      EXOGENOUS EVENT: ID=EV2, TIME=PROC(TIME2);                                00014650  1465                 2
      EXOGENOUS EVENT: ID=EV3, TIME=PROC(TIME3);                                00014660  1466                 3
                                                                                00014670  1467
      PROCEDURE: NAME=TIME1, TYPE=SUBROUTINE;                                   00014680  1468                 4
C     THIS MODEL ASSUMES-                                                       00014690  1469                 5
C       AN AVERAGE OF 8 TRANSACTIONS PER MONTH PER SOLDIER AND A DATA           00014700  1470                 6
C       BASE OF 203 SOLDIERS.  THE CHARACTERIZATION OF EACH                     00014710  1471                 7
C       TRANSACTION IS GIVEN IS THE 'TPARM' ARRAY. THE AVERAGE STAY             00014720  1472                 8
C       PER SOLDIER IS ASSUMED TO BE 6 MONTHS                                   00014730  1473                 9
C                                                                               00014740  1474                10
C     GIVEN THESE ASSUMPTIONS, THIS EXO SERVICE-                                00014750  1475                11
C       IS INVOKED ONCE EVERY 5.89 MINUTES AND STARTS A RANDOMLY                00014760  1476                12
C       SELECTED TRANSACTION ACCORDING TO THE PROBABILITY DISTRIBUTION          00014770  1477                13
C       FUNCTION GIVEN IN THE 'POINTS' ARRAY IN 'START' EXO SERVICE.            00014780  1478                14
C                                                                               00014790  1479                15
C     NOTE- THE DATA USED IN CALCULATING THE TRANSACTION ARRIVAL TIMES          00014800  1480                16
C       WAS BASED ON 20 WORKING DAYS PER MONTH AND 8 HOURS OF TERMINAL          00014810  1481                17
C       INPUT PER DAY.                                                          00014820  1482                18
      COMMON /SYSTIM/ $CLOCK,$CLK1,CLK2                                         00014830  1483                19
      COMMON /SYSINF/ SYSINF                                                    00014840  1484                20
      INTEGER $UNFMI,I,SEED,SYSINF(40)                                          00014850  1485                21
      REAL RN,$PWLIN,$UNIFM,POINTS(6,2)                                         00014860  1486                22
      REAL*8 ITIME,RP                                                           00014870  1487                23
      REAL*8 $CLOCK,$CLK1,$CLK2                                                 00014880  1488                24

........10........20........30........40........50........60........70........80
```

```
                 INFORMATION PROCESSING SYSTEM SIMULATOR                                PAGE  63
                      STANDARD INPUT STREAM LISTING
                             DATE ... 09/10/77
                   INPUT CARD IMAGE                                  INPUT   ALT INPUT   MEMBER
.........10........20........30........40........50........60........70........80  SEQ-NO  LEV SEQ-NO  REF-NO

        DATA ITIME/0.0D+00/                                          00014890   1489        25
        DATA SEED/1/                                                 00014900   1490        26
        DATA POINTS/ 0.0, 1.0, 2.0, 3.0, 4.0, 5.0,                   00014910   1491        27
     1              0.0, 0.2, 0.5, 0.7, 0.8, 1.0/                    00014920   1492        28
                                                                     00014930   1493
C       IF(ITIME .NE. 0.) GO TO 20                                   00014940   1494        30
        DO 10 I=1,33                                                 00014950   1495        31
   10     SYSCOM(I)=0                                                00014960   1496        32
   20 CALL $SREAL(ITIME,SYSCOM,34)                                   00014970   1497        33
C       ONE TRANSACTION ARRIVES EVERY 5.89 MINUTES +-5 MIN           00014980   1498        34
C           ( = 1 TRANS EVERY 353635 MSEC)                           00014990   1499        35
C                                                                    00015000   1500        36
        I = $UNFMI(1,2,SEED)                                         00015010   1501        37
C       CALCULATE THE DEVIATION (MIN IS -5 MIN, MAX IS +5 MIN)       00015020   1502        38
        RN = $UNFM(0.0, 1.0, 1)                                      00015030   1503        39
        RP = $PWLIN(POINTS,6,RN)                                     00015040   1504        40
C       CONVERT DEVIATION MINUTES TO MSEC                            00015050   1505        41
        RP = RP * 60000.                                             00015060   1506        42
C       NEXT ARRIVAL TIME = CURRENT TIME + 5 MIN +- THE DEVIATION    00015070   1507        43
        IF(I .EQ. 1) ITIME =  353.635D+03 + RP                       00015080   1508        44
        IF(I .EQ. 2) ITIME =  353.635D+03 - RP                       00015090   1509        45
        RETURN                                                       00015100   1510        46
        END: PROCEDURE;                                              00015110   1511        47
                                                                     00015120   1512

.........10........20........30........40........50........60........70........80
```

```
                         INFORMATION PROCESSING SYSTEM SIMULATOR                                  PAGE   64
                              STANDARD INPUT STREAM LISTING
                                    DATE ... 09/10/77
                  INPUT CARD IMAGE                                       INPUT    ALT INPUT  MEMBER
.........10........20........30........40........50........60........70........80   SEQ-NO   LEV SEQ-NO  REF-NO

      PROCEDURE: NAME=TIME2, TYPE=SUBROUTINE;                   00015130    1513               48
C     THIS PROCEDURE ESTABLISHES THE TIME AT WHICH THE PAGE     00015140    1514               49
C     CACHE STATISTICS ARE TO BE PRINTED                        00015150    1515               50
        REAL*8 ITIME                                            00015160    1516               51
        DATA ITIME/7.200001D+06/                                00015170    1517               52
        CALL $SREAL(ITIME,SYSCOM,34)                            00015180    1518               53
        RETURN                                                  00015190    1519               54
      END: PROCEDURE;                                           00015200    1520               55
                                                                00015210    1521
      PROCEDURE: NAME=TIME3, TYPE=SUBROUTINE;                   00015220    1522               56
C     THIS PROCEDURE ESTABLISHES THE TIME AT WHICH A SNAPSHOT   00015230    1523               57
C     OF THE STANDARD IPSS STATISTICS ARE TO BE PRINTED         00015240    1524               58
        REAL*8 ITIME                                            00015250    1525               59
        DATA ITIME/7.2D+06/                                     00015260    1526               60
        CALL $SREAL(ITIME,SYSCOM,34)                            00015270    1527               61
        RETURN                                                  00015280    1528               62
      END: PROCEDURE;                                           00015290    1529               63
                                                                00015300    1530
    END: EXOGENOUS EVENT STREAM;                                00015310    1531               64
                                                                00015320    1532
                                                                00015330    1533
    DEFINE MODEL: ID=SIOPRS,                                    00015340    1534
         COMPONENTS=(CF(SYSTEM.CF),FX(SYSTEM.FX));              00015350    1535
                                                                00015360    1536

.........10........20........30........40........50........60........70........80
```

```
                     INFORMATION PROCESSING SYSTEM SIMULATOR                                   PAGE   65

                          STANDARD INPUT STREAM LISTING

                               DATE ... 09/10/77

                     INPUT CARD IMAGE                                          INPUT    ALT INPUT   MEMBER
........10........20........30........40........50........60........70........80     SEQ-NO  LEV SEQ-NO  REF-NO

    EQUATE: CF=START,  EX=EV1;                                        00015370    1537                  2
    EQUATE: CF=BUFST,  EX=EV2;                                        00015380    1538                  3
    EQUATE: CF=SNAPST, EX=EV3;                                        00015390    1539                  4
                                                                      00015400    1540
    SIMULATE: STREAM=START, TRACE=OFF, STOP=(TIME=28800010.),         00015410    1541                  6
              ERROR=(TERMINATE=10, STATISTICS=NO);                    00015420    1542                  7
                                                                      00015430    1543
   END: MODEL;                                                        00015440    1544                  9
```

APPENDIX C

RESULTS OF THE RESEARCH ACTIVITIES

This appendix contains the conclusions drawn from the research activities. These conclusions are based in a large part from the experiences gained with the IPSS SIDPERS/IDMS modeling activities. Recall that the purpose of the modeling activities was to demonstrate the feasibility and suitability of the Information Processing System Simulator (IPSS) with respect to modeling large complex information systems. The system modeled to demonstrate these capabilities was a host/backend processor configuration which supported interactive application processing and a data base management system. The modeled software processes included a detailed characterization of application loading, DBMS processing, and the operating system functions of task management, job scheduling and resource allocation.

The research proposal identified 164 different extensions to the then existing IPSS. As a result of this research, 139 new language and statistics gathering features have been incorporated into the IPSS. Other extensions not in the original proposal were also identified as useful features and are being added to the IPSS. The results are reported in the following three sections of this appendix:

Section	Area Covered
C1	Data Base Systems - specific extensions
C2	Information System - related performance statistics
C3	Computer Networks - specific extensions

The vehicle used for assessing the usefulness of the suggested IPSS extensions was to model the complex systems architecture described in previously in this report. For evaluative purposes, the proposed extensions to the IPSS are each assigned one of the following assessment dispositions:

Assessment Conclusion	Category/Disposition
Not Examined	NE
Examined and Rejected	ER
Examined and found to be useful	
- implemented as proposed	EU-I
- implemented but merged into another facility.	EU-IM
- defined and not implemented	EU-D
- defined and not implemented but used in paper model	EU-DP

Dispositions D and DP are intended to show the depth to which an IPSS language statement under consideration has been examined. Paper models demonstrate effectively the definitional capability of IPSS to characterize information system activities of interest, and as such are a firm indication of the desire of implementing the statement into IPSS.

C1. IPSS LANGUAGE STATEMENTS FOR CHARACTERIZING DATA BASE SYSTEMS

The IPSS system was designed to analyze information processing systems in their totality. To date, IPSS mechanisms for the definition and evaluation of hardware, computer system software and storage level data structures have been extensively developed and tested. The next phase in the continued development of IPSS was the incorporation of features for the description, manipulation and evaluation of logical data structures of the kind normally encountered in modern data base systems.

The goal for this aspect of the research was to identify the usefulness of incorporating a set of generalized data base-related characterization facilities into the IPSS. Data Base Systems (DBS) characterization is provided through two components: one definitional in nature which is specifically designed to represent logical data structures, and the other procedural to represent application and DBS

software processing. These components have been named the Data Base Structure component and the Data Base Access component, respectively. The relation of these two components to the existing model architecture is shown in Figure C1-1. For purposes of identification they have been titled the IPSS/DBS submodel architecture. The proposed language for characterizing DBS facilities for the IPSS/DBS are identified in Table C1-1. Also shown are the results of their evaluation as part of this research effort.

C1.1 The Data Base Structure Component

The purpose of the Data Structure component is to provide the modeler with a set of facilities which allows one to define logical data structures and to characterize the relationships among them. This component is defined independently of the data base access component (See Section C1.2) and thus these facilities can be applied to a variety of DBS architectures and application environments.

The Data Base Structure component facilities serve the following functions:

1. They define templates for schemas consisting of record types and sets, instances of which can be transient entity definitions in the DBA component;
2. They allow data base structures to be viewed as a collection of contiguous logical areas and to map these onto non-contiguous domains of a DBS's storage structure and also onto other logical data structures;
3. They specify allocation policies of record type records to the sets in which they participate; and
4. They allow the characterization of access paths between record types of a schema.

The Data Base Structure component permits the modeler to investigate the effects on system behavior caused by alternate set, record type, and access path definitions. The definitional facilities provided allow the modeler to investigate a wide spectrum of logical data structure organizations and allocation policies.

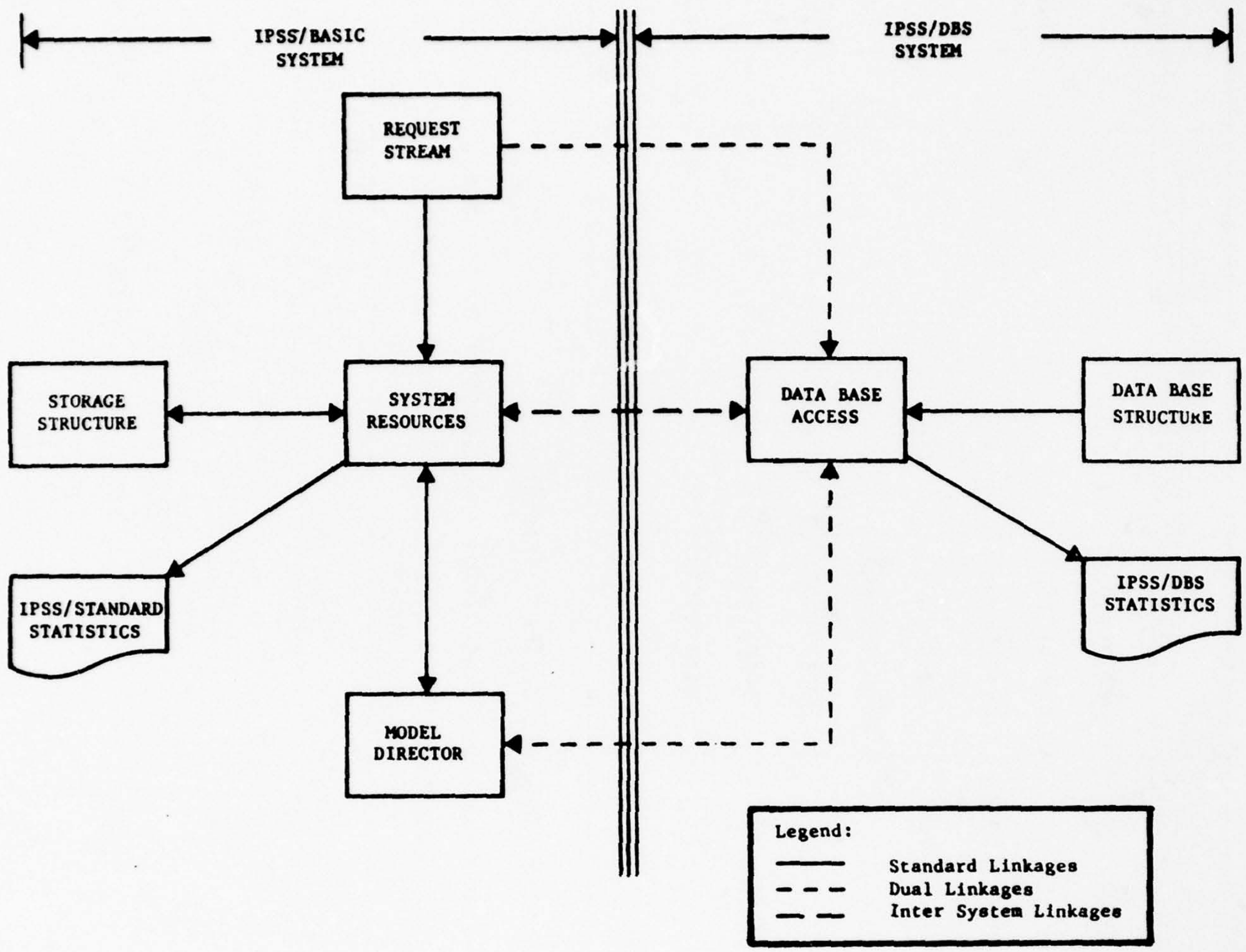


Figure C1-1 IPSS System Architecture with DBMS Extensions

Table C1-1. IPSS/DBS Facilities

Statement	Type	Status in IPSS		Disposition*
		Current	Additional	
BUFFER POOL	Definitional	X		
DML SERVICE	Executional		X	EU-I
ENDOGENOUS SERVICE	Executional	X		
EXOGENOUS SERVICE	Executional	X		
EXTENT	Definitional		X	EU-IM
REALM	Definitional		X	EU-I
RECORD TYPE	Definitional		X	EU-I
SCHEMA	Definitional		X	EU-I
SET TYPE	Definitional		X	EU-I

*Disposition codes are defined as follows:

EU-I : Examined, found Useful - Implemented as proposed;

EU-IM: Examined, found Useful - Implemented by Merging into another facility.

Schema Definition

The DEFINE SCHEMA - END SCHEMA statement sequence defines a logical data structure of inter-related elements. Within this sequence are the definitions of sets, record types and extents.

Set Definition

The SET TYPE statement characterizes connections between record types in a schema. This statement identifies a collection of record types, names the collection, specifies the type of linkage between the record types, identifies the type of participation (either owner or member) of the record types in the set, and characterizes the relative frequency of set occurrences.

Record Type Definition

The RECORD TYPE statement characterizes those properties of a record type which are independent of secondary storage. The physical attributes are expressed in terms of the record format, the number of record occurrences, and record origin. An association with an extent is also provided for linearization. These physical characteristics are independent of the record location in secondary storage. (The IPSS Storage Structure component performs the required mappings from logical records to storage structure representation.) This Data Base Structure component facility completes the record type definition of the Data Base Access component whose major purpose is to define the source of the records.

Extent Definition

The EXTENT statement defines one or more extent facilities. The extent provides the interface between a record type's logical address space (defined via the RECORD TYPE statement) and the Realm specification defined via the REALM statement. Any number of extents may be defined.

Realm Definition

The REALM facility is used to linearize the address space of a schema. The REALM statement provides the external identifier to the REALM facility and provides the space management area associated with the REALM. Record types may be assigned to realms through the EXTENT statement. As many realms as desired may be defined for each schema definition. IPSS views a realm to be the equivalent of a virtual data set.

C1.2 The Data Base Access Component

The Data Base Access component is the central component in an IPSS/DBS defined model. Within it are contained the definitions of all the required DBS and application level software. Hardware resources are limited to main memory buffers used by the DBS software and user work areas. All DBS-related entity type facilities are defined within this component. This subsection describes those facility definition statements which can appear within a Data Base Access component definition. These represent extensions to the basic IPSS language.

In addition to the facilities identified below, the modeler can include any number of standard IPSS procedure facilities and Fortran Subprogram definitions. These defintions can appear anywhere in a Data Base Access component definition with the exception that they cannot be imbedded within any other facility definition. This implies that a Procedure or Subprogram facility definition cannot appear within:

1. Another Procedure,
2. Another Subprogram,
3. An Endogenous Service, or
4. An Exogenous Service.

They may, however, be referenced from within any of the above.

Procedure and Subprogram syntax and semantics are discussed in Volume I of the IPSS Snytax & Semantics document (Section 3.6).

DML Service Definition

The DML SERVICE - END DML SERVICE statement sequence corresponds in function to the standard IPSS Endogenous Service definition. It has been named DML service in the DBA component to more accurately reflect DBS service request processing.

Record Type Definition

The RECORD TYPE statement defines a record type facility within the Data Base Access Component and identifies the source of the records which are created for the specified record type facility.

Schema Definition

The SCHEMA statement identifies a schema facility to the Data Base Access component. The named Schema is associated with a schema definition in the Data Structure component through EQUATE statements in the Model Stream Component.

C1.3 IPSS/DBS Oriented Built-In Facilities

IPSS/DBS Built-in Procedure Facilities provide the modeler with a large number of predefined capabilities whose purpose is to reduce the time needed to develop data base management system models. These procedural facilities free the modeler from the detailed mechanics of the simulation process. The built-in procedures have been grouped into three categories based on features or functions common to each group they are:

1. I/O-oriented procedures,
2. Schema-oriented procedures, and
3. Data structure traversal-oriented procedures.

Table C1-2 contains an alphabetized listing of all the IPSS/DBS Built-in procedures and denotes the dispositions of the statements. All Built-in procedure statements are converted to Fortran subroutine calls with the parmeters associated with that statement transposed to subprogram parameters. Formal descriptions for these IPSS facilities

Table C1-2 IPSS/DBS Built-in Procedures

Statement	Type	Status in IPSS Current	Additional	Disposition
ACQUIRE RECORD	Executional		X	EU-DP
ALLOCATE SCHEMA EXTENT	Executional		X	EU-IM
ALTER SET	Executional		X	NE
CONFIGURE CEQ	Executional	X		
COPY SET	Executional		X	NE
COPY ROUTE	Executional		X	EU-DP
CREATE ROUTE	Executional		X	EU-I
CREATE SCHEMA	Executional		X	EU-I
CREATE SCHEMA EXTENT	Executional		X	EU-IM
CREATE TRANSACTION	Executional	X		
DESTROY ROUTE	Executional		X	EU-I
DESTROY SCHEMA	Executional		X	EU-I
DESTROY SCHEMA EXTENT	Executional		X	EU-IM
DISABLE EVENT	Executional	X		
DO TO LABEL	Executional	X		
ENABLE EVENT	Executional	X		
END DO	Executional	X		
FIND IN QUEUE	Executional	X		
FIND MEMBER	Executional		X	EU-I
FIND NEXT BUFFER	Executional	X		
FIND ROUTE	Executional		X	EU-DP
FREE BUFFER	Executional	X		
GET BUFFER	Executional	X		
GET SAVE AREAS	Executional	X		
IDENTIFY OWNER	Executional		X	EU-DP
IDENTIFY SET OCCURRENCE	Executional		X	EU-DP
INITIATE TASK	Executional	X		
MERGE QUES	Executional	X		
MODIFY BUFFER POOL	Executional	X		
MODIFY QUEUE	Executional	X		
MODIFY REALM	Executional		X	EU-DP
MODIFY ROUTE	Executional		X	EU-DP
MODIFY SCHEMA EXTENT	Executional		X	EU-IM
MODIFY SCHEMA RECORD TYPE	Executional		X	EU-DP
MODIFY SCHEMA SET	Executional		X	EU-DP
PLACE IN QUEUE	Executional	X		
POST SEMAPHORE	Executional	X		
POST SIGNAL	Executional	X		
PROCESS TIME	Executional	X		
PUT SAVE AREAS	Executional	X		
RELEASE SCHEMA EXTENT	Executional		X	EU-IM
REQUEST DML SERVICE	Executional		X	NE
REQUEST SERVICE	Executional	X		
RESUME SCAN	Executional	X		
SET FACILITY STATUS	Executional	X		
SET INDIRECT REFERENCE	Executional	X		

Table C1-2 IPSS/DBS Built-in Procedures (Continued)

Statement	Type	Status in IPSS Current	Status in IPSS Additional	Disposition
SET PRIORITY	Executional	X		
SET SYSTEM PARAMETER	Executional	X		
START QUEUE STATISTICS	Executional	X		
START USAGE STATISTICS	Executional	X		
STOP QUEUE STATISTICS	Executional	X		
STOP SIMULATION	Executional	X		
STOP USAGE STATISTICS	Executional	X		
STORE RECORD	Executional		X	NE
TERMINATE SERVICE	Executional	X		
TERMINATE TASK	Executional	X		
WAIT DML SERVICE COMPLETE	Executional		X	NE
WAIT FACILITY STATUS	Executional	X		
WAIT RECORD	Executional		X	NE
WAIT SEMAPHORE	Executional	X		
WAIT SERVICE COMPLETE	Executional	X		
WAIT SIGNAL	Executional	X		

*EU-I : Examined, found Useful - Implemented as proposed

EU-IM: Examined, found Useful - Implemented by Merging into another facility

EU-DP: Examined, found Useful - defined and not implemented but used in paper model

NE : Not examined.

are not presented in this report. IPSS Syntax and Semantics, Vols. I and II presents the following items of information for each Built-in Procedure:

1. Statement syntax,
2. The function of the statement,
3. Semantics for the statement's parameters,
4. A narrative description of the execution mechanism,
5. IPSS generated Fortran source code which replaces IPSS statements and
6. Examples of the use of the statement.

C1.3.1 IPSS/DBS I/O Related Built-in Procedures

The following Built-in Procedures are used to initiate I/O activity and to wait for the I/O to be completed. The ACQUIRE and STORE RECORD procedures determine the source of the record from the DBA component record type facility definition. If a service is specified as the source, then the service is automatically invoked. WAIT RECORD statements are defined to enable service processing to be suspended pending the I/O operation. This section also contains the definition of the statement for initiating and waiting for a DML Service. These procedures can be referenced from within either Procedure facilities or Exogenous and Endogenous Service facilities.

Acquire and Store Record

The ACQUIRE and STORE RECORD Built-in Procedures invoke a service which simulates the acquisition and storage of the specified record to and from the service in which the ACQUIRE or STORE statement appears. The identity of service to be invoked is contained in the definition of the specified record type facility in the DBA component.

Wait Record

The function of the WAIT RECORD Built-in Procedure is to place the specified transaction into the wait state unitl the completion of the Endogenous service which the specified transaction invoked via an ACQUIRE or STORE RECORD statement. Service for the suspended transaction resumes after the specified wait criteria (count and request number) are satisfied. Each completed Endogenous Service routine satisfying the request number criterion decrements the suspended transaction's Wait Count by one and the delayed transaction is reactivated when the count reaches zero and is returned to the active state.

Request DML Service

The REQUEST DML SERVICE Built-in Procedure is used to initiate the execution of a DML Endogenous Service.

Wait DML Service Complete

The function of the WAIT DML SERVICE COMPLETE Built-in Procedure is to place the named transaction into the wait state until the completion of the specified Endogenous DML Service which it invoked. Service for suspended transaction resumes after the specified wait criteria (count and request number) are satisfied. Each completed Endogenous DML Service routine satisfying the request number criterion decrements the suspended transaction's wait count by one and the delayed transaction is reactivated when the count reaches zero and is returned to the active state.

C1.3.2 IPSS/DBS Schema Related Built-in Procedures

The following Built-in procedures are used to change the current state of facilities defined in the DATA Structure component of an IPSS/DBS model. These procedures can be referenced from within either Procedure facilities or Exogenous and Endogenous Service facilities.

Allocate and Release Schema Extent

The effect of executing the ALLOCATE SCHEMA EXTENT Built-in Procedure is to simulate the allocation of a logical address space to the Extent subfacility. The allocation causes the extent to take on a displacement attribute which specifies an extent's displacement relative to a Realm facility. This is done by placing the calculated displacement of the extent start into the Schema facility's extent table. The displacement is a real number whose dimension is in terms of the reference units implied for the Realm. This information is used by the Acquire and Store Record Built-in procedures (when the desired record type has an origin of Realm) to determine the logical address of a record type record. If a Schema Extent is not allocated, it is assumed to have zero capacity relative to its ability to hold schema records. The RELEASE SCHEMA EXTENT Built-in Procedure returns allocated address space.

Create and Destroy Schema

The purpose of the CREATE SCHEMA Built-in Procedure is to create for the named Schema facility a copy of its associated Data Structure component definition. The copy is placed in a pre-specified area of the Transaction Area. Once copied, simulation functions involving the Schema facility can occur. The definition remains valid until the issuing of a DESTROY SCHEMA statement for the same Schema facility.

Create and Modify Schema Extent

The function of the CREATE AND MODIFY SCHEMA EXTENT Built-in Procedure is to alter the current attributes associated with a Schema Extent facility. The attributes are first assigned via the extent definition in the Schema facility definition of the Data Structure component. A Schema Extent once created can only be modified or destroyed and can only be "created" when it is not in a "created" state. The procedure can only be invoked after a Schema has been created.

Destroy Schema Extent

The DESTROY SCHEMA EXTENT Built-in Procedure deletes the specified schema extent definition from the named Schema facility. The destroy procedure releases the logical address space associated with the extent by setting its size attribute to zero and removing its Realm facility association.

Modify Realm

The MODIFY REALM Built-in Procedure enables the modeler to change the attributes associated with a Realm facility definition. The procedure changes only those attributes explicitly identified in the Modify Realm Statement.

Modify Schema Record Type

The MODIFY SCHEMA RECORD TYPE Built-in Procedure changes the parameter values of the specified record type facility definition. The original parameters are specified in the RECORD TYPE statements in both the Data Structure and DBS components. The procedure changes only those attributes explicitly identified in the MODIFY RECORD TYPE statement.

Modify Schema Set

The function of the MODIFY SCHEMA SET Built-in Procedure is to modify parameters associated with a Schema's Set subfacility. The original parameters are specified via the Data Structure component's Set facility definition. The procedure changes only those attributes explicitly identified in the MODIFY SCHEMA SET Statement.

C1.3.3 IPSS/DBS Data Structure Trversal Oriented Built-in Procedures

The following Built-in Procedures are used a) to create routes or individual sets based upon the schema definitions in the Data Structure component, b) to modify routes or sets by specifying the elements to be changed, and c) to destroy an existing route. The procedure statements are

divided into two groups, those which operate on sets, and those which operate on routes. These procedures can be referenced from within either Procedure facilities or Endogenous and Exogenous Service facilities.

Copy Set

The COPY SET Built-in Procedure creates a set occurrence from the specified set definition. The set occurrence is placed into an output array which the modeler designates. The set may either be copied from the corresponding set definition in the Data Structure component or from an existing route. In either case the modeler may change the set traversal currency indicator during the copy process.

Alter Set

The ALTER SET Built-in Procedure modifies a previously created set occurrence. The specified set occurrence may either be the output area of a previously executed Copy Set procedure or may exist within a route. This statement can be used to change the owner and member record types of the set as well as the record occurrences for these record types.

Find Member

The FIND MEMBER Built-in Procedure identifies a member record occurrence of a set relative to the specified set traversal currency. The output of this procedure is an updated member record occurrence and/or member record type of the set's traversal currency. A NO MEMBER EXIT parameter is provided for those cases in which it is not possible to update this currency; for example, FIND NEXT MEMBER when the number of occurrences associated with the member record type is zero. The FIND MEMBER examines the set and the record type facility definitions in the Data Structure component in order to determine the existence of the desired record. The statement types are as follows: a) FIND FIRST MEMBER specifies that the first member record occurrence for the designated set type is to be located and its identity placed in the set's traversal currency. By "identity" we mean the member record type, the

record occurrence within the set and the record occurrence within the record type. The success of this operation depends on a data structure which directly links the set traversal currency before execution of this procedure to the first record in the set; b) FIND PRIOR MEMBER specifies that the record to be identified is the record occurrence which immediately precedes the one specified by the set traversal currency. The success of this option likewise depends upon a direct link between the record in the set traversal currency and the prior record occurrence; c) FIND NEXT MEMBER specifies that the record to be identified is the record occurrence which immediately follows the record occurrence of the set traversal currency. The successful completion of this procedure depends upon the existence of a direct link between the record specified by the set traversal currency and the next record occurrence in the set; d) FIND LAST MEMBER specifies that the last record occurrence for the specified set type is to be located and its identity placed in the set's traversal currency. The success of this option depends upon a data structure which directly links this record to the set traversal currency before execution of this procedure.

Identify Owner

The IDENTIFY OWNER Built-in Procedure identifies the record type which is the owner of the specified record type and identifies the set in which they both participate. This procedure examines the set and record type facility definitions in the Data Structure component of the current schema to determine this relationship. All the sets which own a given record type can be obtained by specifying IDENTIFY OWNER followed by successive IDENTIFY NEXT OWNER statements.

Identify Set Occurrences

The IDENTIFY SET OCCURRENCE Built-in Procedure identifies the sets which the current member record occurrence owns. By current member record occurrence we mean the traversal currency as reflected in the

specified set. This procedure examines the set and record type facility definitions in the Data Structure component of the current schema to determine ownership of a non-empty set of records. All of the sets owned by a record can be obtained by specifying IDENTIFY SET OCCURRENCE followed by successive IDENTIFY NEXT SET OCCURRENCE statements.

Create Route

The CREATE ROUTE Built-in Procedure determines the existence of a path between two record types of a schema. The beginning and ending nodes of the path are specified in terms of record type identifiers. The function of this procedure is to determine if these two record types are connected and if the connections meet all the set constraints. The set declarations of the identified schema are examined in the order of their declaration until either a path is found or all the declarations for the schema have been explained. All the paths between two record types can be located by specifying PATH = FIRST followed by successive PATH = NEXT statements. When a path is found, its sets are placed in the $TRANS array beginning with the location returned as the value of the route identifier. If no path is found, the value of the route identifier will be a zero.

Destroy Route

The DESTROY ROUTE Built-in Procedure deletes the specified route from the $TRANS array. This allows the re-use of the space occupied by the route for other modeling activities. The execution of this procedure does not affect schema definitions in the Data Structure component.

Modify Route

The MODIFY ROUTE Built-in Procedure changes the route definition by either: a) reestablishing the current set, b) deleting or inserting sets, or c) concatenating another route to the existing one. A route consists of a sequence of set traversal tables. This procedure can change this

sequence by deleting, inserting or concatenating set traversal tables relative to any of the sets in the route. Only one type of change can be specified for each Modify Route Statement.

Copy Route

The COPY ROUTE Built-in Procedure creates a route and copies the specified route data into it. The sets in the output route can be modified during this process. These modifications consist of: a) re-establishing the current set, b) inserting or deleting sets, or c) concatenating another route to the output route. These modifications can be specified either by naming specific sets or by referencing the sets in the route relative to the set currency.

Find Route

The FIND ROUTE Built-in Procedure uses a previously created route and the current schema definition to establish specific record occurrences within the sets of the route. A set other than the first set of the route may be designated as the beginning set for the procedure to establish record occurrences. A set other than the last set in the route may also be specified.

C2. IPSS PERFORMANCE MEASURES

The proposal identified 120 extensions to the currently available model statistics. The purpose of this section of Appendix C is to present the results of the research in the area of added statistical outputs from IPSS models.

The Information Processing System Simulator provides a modeler with a number of statistics concerning the behavior of modeler defined entities and IPSS supplied built-in services. These statistics fall into eight general categories listed below. The evaluative results of the research is found in Subsections C2.2 through C2.8 which correspond to these statistics categories:

1. Operational Statistics,
2. Request Stream Statistics,
3. I/O Activity,
4. Queueing Statistics,
5. Utilization Statistics,
6. Wait Statistics,
7. Service Statistics, and
8. Task/Activity Statistics.

In addition to these "automatic" statistics, the modeler can employ the complete facilities of the Fortran language to develop his own statistics. Statistics are printed automatically at the conclusion of each model simulation unless explicitly inhibited.

Time constraints prohibited the implementation of any new statistics. However, definitions for a number of the proposed statistics were completed and are reported in the appropriate subsection. These statistics employ both IPSS facility specific variables and those global to an entire model. For each defined statistic the variables required to support their collection and calculation are also discuss along with the method for data collection including the specific IPSS statements responsible for the activity. Note: all variable names used in these definitions are for publication purposes only.

C2.1 IPSS Operational Statistics

These statistics provide an overview of the operational characteristics of the current simulation execution. The Clock and Termination Statistics involve the length of the run, in terms of both simulation time and CPU time, and the reason for termination. The Transaction Statistics provide insight into the degree of transaction activity generated by the current simulation run. Statistics on the number of transactions generated and the average life span of a transaction are collected by transaction type. The results of their evaluation are shown in Table C2.1-1. Table C2.1-2 identifies the required data to meet the definitional requirements for these statistical extentions to the IPSS. The explicit statistic definitions are shown in Table C2.1-3.

Table C2.1-1 IPSS Operational Statistics

Statistic Type	Specific Statistic	Currently Available	Proposed Extention	Disposition*
Clock	Absolute elapsed simulation time	X		
	Elapsed simulation time since last reset	X		
	Elapsed simulation time of current model execution	X		
	Elapsed CPU time (IPSS run time)		X	EU-DP
	Scaling factor		X	EU-DP
Termination	Number of errors	X		
	Termination status		X	EU-DP
Transaction (By Transaction type)	Total number generated		X	EU-DP
	Current number in system	X		
	Maximum number in system	X		
	Starting number in system	X		
	Total number terminated		X	EU-DP
	Mean time of existence		X	EU-DP

*Distribution codes used in this table are defined as follows:

DU-DP: Examined, found useful - Defined, not implemented but used in paper models

Table C2.1-2 IPSS Operational Statistics - Required Model Statements

Variable Name	Description of the Variable	Method of data Collection
VARIABLES GLOBAL TO AN IPSS MODEL		
CLOCK1	Absolute elapsed simulation time	Automatically by simulation driver
CLOCK2	Elapsed simulation time since last reset	Automatically by simulation driver
CLOCK3	Elapsed simulation time of current model execution	Automatically by simulation driver
CPUTIM	Beginning CPU time of current IPSS run	SVC call by simulation driver
SCALEF	Scaling factor	Specified by user
ERRM	Total number of errors in current model execution	Updated by internal IPSS error routine
TERMST	Termination status	Specified by user
VARIABLES WHICH ARE TRANSACTION SPECIFIC		
$TNGEN_A$	Total number generated	Updated by simulation driver upon generation of new transaction
$TNCUR_A$	Current number in system	Maintained by driver in SYSINF
$TNMAX_A$	Maximum number in system	Either user specified or system default
$TNSTRT_A$	Starting number in system	Either user specified or system default
$TNTRM_A$	Total number terminated	Updated by driver upon termination

Table C2.1-3 IPSS Operational Statistics - Definition

Stastic Type	Specific Statistic Name	Definition
Clock	Absolute elapsed simulation time	CLOCK1
	Elapsed simulation time since last reset	CLOCK2
	Elapsed simulation time of current model execution	CLOCK3
	*Elapsed CPU time (IPSS run time)	CPUTIM
	*Scaling Factor	SCALEF
Termination	Number of errors	ERRM
	*Termination Status	TERMST
Transaction (by transaction type)	*Total number generated	$TNGEN_A$
	Current number in system	$TNCUR_A$
	Maximum number in system	$TNMAX_A$
	Starting number in system	$TNSTRT_A$
	*Total number terminated	$TNTRM_A$
	*Mean time of existence	$(CLOCK3/TNGEN_A)$

*Statistic currently exists in IPSS models.

C2.2 IPSS Request Stream Statistics

These statistics describe the behavioral characteristics of the Request Stream facilities. For each Exogenous Event a profile of the interarrival time mechanism is presented. The event of the IPSS built-in Endogenous Services, ENABLE/DISABLE Event, on the arrival of events is also summarized. The collection of these statistics is an automatic by-product of the simulation drive mechanism. Table C2.2-1 presents the evaluation results. Table C2.2-2 identifies the required model data and Table C2.2-3 the actual statistic definition.

Table C2.2-1 IPSS Request Stream Statistics

Statistic Type	Specific Statistic	Currently Available	Proposed Extension	Disposition*
General	Number generated	X		
	Number terminated		X	EU-DP
Arrival Time	First generated arrival time		X	EU-DP
	Average interarrival time		X	EU-DP
	Last arrival time generated		X	EU-DP
Enable/Disable	Number of enables		X	EU-DP
	Total time enabled		X	EU-DP
	Average enable time		X	EU-DP
	Number of disables		X	EU-DP
	Total Time Disabled		X	EU-DP
	Average disable time		X	EU-DP

*Disposition Codes used in this table are defined as follows:

EU-DP: Examined, found useful - Defined, not implemented but useful in paper models

Table C2.2-2

IPSS Request Stream Statistics - Required Model Statements

Variable Name	Description of the Variable	Method of Data Collection
EXGEN	Number generated	Updated by driver
EXTRM	Number terminated	Updated by driver
EFARR	First generated arrival time	Specified by user (either directly or via Procedure)
ELARR	Last (latest) arrival time generated	Updated by driver
ENEN	Number of enables	Updated by ENABLE statement
ETLEN	Time of last enable/disable	Updated by both the ENABLE and DISABLE statements
ETTEN	Total time enabled	Calculated by DISABLE statement as (CLOCK3 - ETLEN) + ETTEN
ENDSA	Number of disables	Updated by DISABLE statement
ETTDSA	Total time disables	Calculated by ENABLE statement as (CLOCK3 - ETLEN) + ETTDSA

Table C.2.2-3
IPSS Request Stream Statistics - Definitions

Statistic Type	Specific Statistic Name	Definition
General	Number Exogenous requests generated	EXGEN
	*Number Exogenous requests terminated	EXTRM
Arrival time	*First generated Exogenous event arrival time	EFARR
	*Mean interarrival time	(ELARR - EFARR)/EXGEN
	*Last arrival time	ELARR
Enable/Disable	*Number of enables	ENEN
	*Total time enabled	ETTEN
	*Mean enable time	(ETTEN/EXEN)
	*Number of disables	ENDSA
	*Total time disabled	ETTDSA
	*Mean disable time	(ETTDSA/ENDSA)

*Statistic currently exists in IPSS models

C.2.3 IPSS I/O Activity Statistics

All of the proposed new statistics in the area of I/O activity were concerned with the concept of locality. Statistics were proposed at the Data Set level and within the Data Set, by File, Extent, Volume, Cylinder and Track, and at the Area level, and within Area by Segment and by Volume. All of these statistics are to be gathered by the GET ADDRESS statement. Major modifications however, will be required to the existing IPSS code supporting the IPSS GET ADDRESS Built-in Procedure in order to gather the required data. The purpose of these statistics is to measure the degree of hardware and file management level I/O activity that occurred during the current model. Measures of I/O routine performance are collected by I/O routine type, by associated access mechanism, and by referenced Data Set and File. Statistics showing the degree of locality of consecutive logical and physical references by Data Set and Area were also proposed.

A second set of statistics of similar intent was proposed for the IPSS/DBS subsystem. Since these statistics are closely associated with other I/O activities, the decision was made to delay the examination of this class of statistics until the IPSS/DBS subsystem was operational. However, the I/O-related statistics are defined in Table C2.3 for report completeness.

Table C.2.3 IPSS I/O Activity Statistics

Statistics Type	Specific Statistic	Currently Available	Proposed Extension	Disposition
I/O Invocations (By I/O Statement)	Number of calls	X		
	Percent zero-time calls	X		
	Mean time per call	X		
	Mean time for non-zero calls	X		
	Standard deviation of time per call	X		
	Standard deviation of time per call (excluding zero-time calls)	X		
	(Mean and standard deviation of various physical attributes, i.e., cylinders crossed, characters read, etc.)	X		
I/O Invocations (By Access Mechanism)	By I/O type for each Access Mechanism:			
	Number of calls	X		
	Number of zero-time calls	X		
	Percent zero-time calls	X		
	Mean routine time	X		
	Mean routine time (excluding zero-time calls)	X		
	Standard deviation of mean routine time	X		
	Standard deviation of mean routine time (excluding zero time calls)	X		
I/O Invocations (By Data Set and File)	By I/O type for each Data Set and File:			
	Number of calls	X		
	Number of zero-time calls	X		
	Percent of zero-time calls	X		
	Mean routine time	X		
	Mean routine time (excluding zero-time calls)	X		
	Standard deviation of mean routine time	X		
	Standard deviation of mean routine time (excluding zero-time calls)	X		

Table C.2.3 IPSS I/O Activity Statistics (Continued)

Statistics Type	Specific Statistic	Currently Available	Proposed Extension	Disposition
Locality (Data Set) (Conditional)	By Data Set:			
	Number of consecutive references for each file		X	NE
	By File:			
	Number of consecutive references for each Extent		X	NE
	By Extent:			
	Number of consecutive references to the same Volume		X	NE
	By Volume:			
	Number of consecutive references to the same cylinder		X	NE
	Average number of cylinders crossed		X	NE
	By Cylinder:			
	Number of consecutive references to the same track		X	NE
	Average number of tracks crossed		X	NE
	By Track:			
	Number of consecutive references to the same block		X	NE
	Average number of blocks crossed		X	NE
Locality (Area) (Conditional)	By Area:			
	Total number of references to area		X	NE
	Number of consecutive references to same area		X	NE
	Average number of consecutive references		X	NE
	Average length between consecutive references in terms of reference units		X	NE
	By Segment:			
	Total number of references to segment		X	NE

Table C.2.3 IPSS I/O Activity Statistics (Continued)

Statistics Type	Specific Statistic	Currently Available	Proposed Extension	Disposition
	Number of consecutive references to same segment		X	NE
	Average number of consecutive references		X	NE
	Average length between consecutive references in terms of reference units		X	NE
	By Volume:			
	Number of consecutive references to same volume		X	NE
	Average number of consecutive references		X	NE
	Average length between consecutive references in terms		X	NE

*Disposition codes used in this table are:

NE: Not evaluated

C.2.4 IPSS Qeueing Statistics

An extensive number of queueing statistics are currently gathered during an IPSS model's execution. Several additional extensions to these statistics were proposed and evaluated. These are now being incorporated into the IPSS statistics generation procedures. Table C.2.4-1 identifies both the existing and proposed statistics and the result of the evaluative effort. The purpose of Table C.2.4-2 is to identify the specific data for these statistics. Table C.2.4-3 provides definitions for the statistics.

Table C.2.4-1 IPSS Queueing Statistics

Statistics Type	Specific Statistic	Currently Available	Proposed Extension	Disposition
Busy Period	Total number of busy periods	X		
	Number of zero time busy periods	X		
	Percent zero time busy periods	X		
	Total busy time	X		
	Percent busy	X		
	Mean length of busy period	X		
	Mean length of busy period (excluding zero time periods)		X	EU-DP
	Standard deviation for non-zero time busy periods		X	EU-DP
	Maximum length of a busy period		X	EU-DP
	Minimum (non-zero) length of a busy period		X	EU-DP
	Length of the current busy period		X	EU-DP
Idle Period	Total number of idle periods	X		
	Number of zero-time idle periods	X		
	Percent zero-time idle periods	X		
	Total idle time	X		
	Percent idle	X		
	Mean length of an idle period	X		
	Mean length of an idle period (excluding zero-time periods)		X	EU-DP
	Standard deviation for non-zero time idle periods		X	EU-DP
	Maximum length of an idle period		X	EU-DP
	Minimum (non-zero) length of an idle period		X	EU-DP

Table C.2.4-1 IPSS Queueing Statistics (Continued)

Statistics Type	Specific Statistic	Currently Available	Proposed Extension	Disposition
Queue Length (During a Busy Period)	Mean length	X		
	Maximum length	X		
	Current length	X		
	Standard deviation of mean length	X		
Queue	Total number of queue entries	X		
	Total number of completed queue entries	X		
	Number of zero-time queue entries	X		
	Percent of zero-time queue entries	X		
	Mean queue transit time	X		
	Mean queue transit (excluding zero-time transit)	X		
	Standard deviation of non-zero time queue transit	X		
	Maximum transit time		X	EU-DP
	Minimum (non-zero) transit time		X	EU-DP

Disposition codes used for this Table are:

EU-DP: Examined, found useful - Defined not implemented but useful in paper model

Table C.2.4-2 IPSS Queueing Statistics - Required Model Statements

Variable Name*	Description of the Variable	Method of Data Collection
QCNCE	Current number of concurrent elements in queue	Update by Queue and Depart Queue Statement
QMXNCE	Maximum number of current concurrent elements in queue	Update by Queue statement Queue
QSNCE	Sum of number of concurrent elements in queue	Updated by Depart Queue statement
QSNCES	Sum of (number of concurrent elements in queue)**2	Updated by Queue and Depart Queue statement
QSQE	Total number of queue elements	Update by Queue statement
QSZQE	Total number of zero-time queue elements	Updated by Depart Queue statements
QSTT	Sum of time per transaction	Updated by Depart Queue statement
QSTTS	Sum of (time per trans)**2	Updated by Depart Queue statement
QCTI	Cumulative Time integral	Updated by Queue/Depart Queue statement
QTQDQ	Time of last occurrence of Queue or Depart Queue invocation	Updated by Queue and Depart Queue statement
QSBP	Total number of busy periods	Updated by Queue statement
QSBPL	Sum of length of busy periods	Updated by Depart Queue statement
QSBPLS	Sum of (length of busy periods)**2	Updated by Depart Queue statement
QSIP	Number of idle periods	Updated by Depart Queue statement
QSIPL	Sum of length of time for each idle period	Update by Queue statement

Table C.2.4-2 IPSS Queueing Statistics - Required Model Statements (Continued)

Variable Name*	Description of the Variable	Method of Data Collection
QSIPLS	Sum of (length of idle periods)**2	Updated by Queue statement
QTBIP	Time of last Busy or Idle period start	Updated by Queue statement and Depart Queue
QSZBP	Number of zerotime busy periods	Updated by Queue statement
QSZIP	Number of zerotime idle periods	Updated by Depart Queue statement
QMNBPL	Minimum non-zero busy period length	Updated by Depart Queue statement
QMXBPL	Maximum busy period length	Updated by Depart Queue statement
QCBPL	Length of current busy period	Updated by Depart Queue statement
QMNIPL	Minimum non-zero idle period length	Updated by Queue statement
QMXIPL	Maximum idle period length	Updated by Queue statement
QCIPL	Length of current idle period	Updated by Queue statement
QMXTT	Maximum transit time per transaction	Updated by Depart Queue statement
QMNTT	Minimum non-zero transit time per transaction	Updated by Depart Queue statement

*All data are stored in the statistics area associated with the designated facility.

Table C.2.4-3 IPSS Queueing Statistics - Definitions

Statistic Type	Specific Statistic	Definition
Busy Period	Total number of busy periods	QSBP
	Number of zerotime busy periods	QSZBP
	%zero time busy periods	(QSZBP/QSBP)*100
	Total Busy time	QSBPL
	%busy	(QSBPL/CLOCK3)*100
	Mean length of busy period	(QSBPL/QSBP)
	Standard deviation of mean length	((QSBP*QSBPLS-QSBPL**2)/ (QSBP*(QSBP-1)))**0.5
	Mean length of nonzero time busy periods	(QSBPL/(QSBP-QSZBP))
	Standard deviation of nonzero time busy periods	(((QSBP-QSZBP)*QSBPLS-QSBPL**2) /((QSBP-QSZBP)*(QSBP-QSZBP-1))) **0.5
	Maximum length of a busy period	QMXBPL
	Minimum non-zero length	QMNBPL
	Length of current busy period	QCBPL
Idle Period	Total number of idle periods	QSIP
	Number of zerotime idle periods	QSZIP
	%zero time idle periods	(QSZIP/QSIP)*100
	Total idle time	QSIPL
	%idle	(QSIPL/CLOCK3)*100
	Mean length of idle period	(QSIPL/QSIP)
	Standard deviation of mean length	(((QSIP-QSIPLS)-QSIPL**2) /(QSIP*(QSIP-1)))**0.5

Table C.2.4-3 IPSS Queueing Statistics - Definitions (Continued

Statistic Type	Specific Statistic	Definition
	Mean length of nonzero time idle periods	(QSIPL/(QSIP-QSZIP))
	Standard deviation of nonzero time idle periods	(((QSIP-QSZIP)*QSIPLS-QSIPL**2) /((QSIP-QSZIP)*(QSIP-QSZIP-1)))**0.5
	Maximum length of an idle period	QMXIPL
	Minimum nonzero length	QMNIPL
	Length of current idle period	QCIPL
Queue Length	Mean length	(QSNCE/QSQE)
	Maximum length	QMXNCE
	Current length	QCNCE
	Standard deviation of mean length	((QSQE*QSNCES-QSNCE**2) /(QSQE*(QSQE-1)))**0.5
Queue	Total number of queue entries	QSQE
	Total number of completed queue entries	QSQE - QCNCE = QSCQE
	Number of zerotime queue entries	QSZQE
	%zerotime queue entries	(QSZQE/QSQE)*100
Transit Time	Mean transit time per transaction	(QSTT/(QSQE-QCNCE))
	Mean nonzero time per transaction	(QSTT/(QSCQE-QSZQE))
	Standard deviation of mean transaction time	((QSCQE*QSTTS-QSTT**2) /(QSCQE*(QSCQE-1)))**0.5

Table C.2.4-3 IPSS Queueing Statistics - Definitions (Continued)

Statistic Type	Specific Statistic	Definition
	Standard deviation of mean non-zerotime transactions	(((QSCQUE-QSZQE)*QSTTS-QSTT**2) /((QSCQE-QSZQE)*(QSCQE-QSZQE-1)))**0.5
	Maximum transit time	QMXTT
	Minimum nonzero transit time	QMNTT

C.2.5 IPSS Utilization Statistics

An IPSS model generates utilization statistics on user defined modeled facilities. Several extensions to these statistics were proposed and evaluated in the research. Table C.2.5-1 identifies all IPSS utilization statistics, whether current or proposed and their desposition after evaluation. Table C.2.5-2 identifies the means for their generation and Table C.2.5-3 contains their definitions.

Table C.2.5-1 IPSS Utilization Statistics

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition
Busy Period	Total number of busy periods	X		
	Number of zero time busy periods	X		
	Percent zero time busy periods	X		
	Total busy time	X		
	Percent busy	X		
	Mean length of busy period	X		
	Mean length of busy period (excluding zero time periods)		X	EU-DP
	Standard deviation for nonzero time busy periods		X	EU-DP
	Maximum length of a busy period		X	EU-DP
	Minimum (non-zero) length of a busy period		X	EU-DP
	Length of the current busy period		X	EU-DP
Idle Period	Total number of idle periods	X		
	Number of zero time idle periods	X		
	Percent zero-time idle periods	X		
	Total idle time	X		
	Percent idle	X		
	Mean length of an idle period	X		
	Mean length of an idle period (excluding zero-time periods)		X	EU-DP
	Standard deviation for nonzero time idle periods		X	EU-DP
	Maximum length of an idle period		X	EU-DP
	Minimum (non-zero) length of an idle period		X	EU-DP
	Length of the current idle period		X	EU-DP
Concurrency	Mean level of concurrency	X		
	Maximum level of concurrency	X		
	Current level of concurrency	X		
	Standard deviation of mean level of concurrency	X		

Table C.2.5-1 IPSS Utilization Statistics (Continued)

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition
Seizure	Total number of seizures	X		
	Total number of completed seizures	X		
	Number of zero-time seizures	X		
	Percent zero-time seizures	X		
	Mean seizure time	X		
	Mean seizure time (excluding zero-time seizures)	X		
	Standard deviation of non-zero time seizures	X		
	Maximum length of a seizure		X	EU-DP
	Minimum (non-zero) length of a seizure		X	EU-DP

Disposition codes used in this table are:

EU-DP: Examined, found useful - defined, not implemented but useful in paper model.

Table C.2.5-2 IPSS Utilization Statistics - Required Model Statement

Variable Name*	Description of the Variable	Method of Data Collection
UCNCS	Current number of Concurrent seizures	Updated by Seize/Release statements
UMXNCS	Maximum number of current concurrent seizures	Updated by Seize statement
USNCS	Sum of number of concurrent seizures	Updated by Seize/Release statement
USNCSS	Sum of (number of concurrent seizures)**2	Updated by Seize/Release statement
USSEZ	Total number of seizures	Updated by Seize statement
USZSEZ	Total number of zero-time seizures	Updated by Release statement
USTT	Sum of time per transaction	Updated by Release statement
USTTS	Sum of (time per trans)**2	Updated by Release statement
UCTI	Cumulative Time Integral	Update by Seize and Release statements
UTSR	Time of last seize or release	Updated by Seize and Release statements
USBP	Total number of busy periods	Update by Seize statement
USBPL	Sum of length of busy periods	Update by Release statement
USBPLS	Sum of (length of busy periods)**2	Updated by Release statement
USIP	Number of idle periods	Updated by Release statement
USIPL	Sum of length of idle periods	Updated by Seize statement
USIPLS	Sum of (length of idle periods)**2	Updated by Seize statement
UTBIP	Time of last Busy or idle period start	Updated by Seize and Release statement

Table C.2.5-2 IPSS Utilization Statistics - Required Model Statement (Continued)

Variable Name*	Description of the Variable	Method of Data Collection
USZBP	Number of zerotime busy periods	Updated by Release statement
USZIP	Number of zerotime idle periods	Updated by Seize statement
UMNBPL	Minimum non-zero busy period length	Updated by Release statement
UMXBPL	Maximum busy period length	Updated by Release statement
UCBPL	Length of current busy period	Updated by Release statement
UMNIPL	Minimum non-zero idle period length	Updated by Seize statement
UMXIPL	Maximum idle period length	Updated by Seize statement
UCIPL	Length of current idle period	Updated by Seize statement
UMXTT	Maximum transit time per transaction	Updated by Release statement
UMNTT	Minimum non-zero transit time per transaction	Updated by release statement

*All data are stored in the statistics area associated with the designated facility.

Table C.2.5-3 IPSS Utilization Statistics - Definitions

Statistic Type	Specific Statistic	Definition
Busy Period	Total number of busy periods	USBP
	Number of zerotime busy periods	USZBP
	%zero time busy periods	(USZBP/USBP)*100
	Total busy time	USBPL
	%busy	(USBPL/CLOCK3)*100
	Mean length of busy period	(USBPL/USBP)
	Standard deviation of mean length	(((USBP-USBPLS)-USBPL**2) /(USBP*(USBP-1)))**0.5
	Mean length of non-zero time busy periods	(USBPL/(USBP-USZBP))
	Standard deviation of non-zero time busy periods	(((USBP-USZBP)*USBPLS-USBPL**2) /((USBP-USZBP)*(USBP-USZBP-1)))**0.5
	Maximum length of a busy period	UMXBPL
	Minimum non-zero length	UMNBPL
	Length of current busy period	UCBPL
Idle Period	Total number of idle periods	USIP
	Number of zero time idle periods	USZIP
	%zero time idle periods	(USZIP/USIP)*100
	Total idle time	USIPL
	%idle	(USIPL/CLOCK3)*100
	Mean length of idle period	(USIPL/USIP)
	Standard deviation of mean length	((USIP*USIPLS-USIPL**2) /(USIP*(USIP-1)))**0.5

Table C.2.5-3 IPSS Utilization Statistics - Definitions (Continued)

Statistic Type	Specific Statistic	Definition
	Mean length of non-zero time idle periods	(USIPL/(USIP-USZIP))
	Standard deviation of non-zero time idle periods	(((USIP-USZIP)*USIPLS-USIPL**2) /((USIP-USZIP)*(USIP-USZIP-1)))**0.5
	Maximum length of an idle period	UMXIPL
	Minimum non-zero length	UMNIPL
	Length of current Idle period	UCIPL
Concurrency	Mean level	(USNCS/USSEZ)
	Maximum level	UMXNCS
	Current level	UCNCS
	Standard deviation of mean level	((USSEZ*USNCSS-USNCS**2) /(USSEZ*(USSEZ-1)))**0.5
Seizures	Total number of seizures	USSEZ
	Total number of completed seizures	USCSEZ = USSEZ - UCNCS
	Number of zero-time seizures	USZSEZ
	%zero-time seizures	(USZSEZ/USSEZ)*100
Transit Time	Mean transit time per transaction	(USTT)/USCSEZ
	Mean non-zero time per transaction	(USTT/(USCSEZ-USZSEZ))
	Standard deviation of mean transaction time	((USCSEZ*USTTS-USTT**2) /(USCSEZ*(USCSEZ-1)))**0.5
	Standard deviation of mean non-zero-time transactions	(((USCSEZ-USZSEZ)*USTTS-USTT**2) /((USCSEZ-USZSEZ)*(USCSEZ-USZSEZ-1)))**0.5

Table C.2.5-3 IPSS Utilization Statistics - Definitions (Continued

Statistic Type	Specific Statistic	Definition
	Maximum transit time	UMXTT
	Minimum non-zero transit time	UMNTT

C.2.6 IPSS Wait Statistics

These statistics measure the degree to which individual transactions were delayed until specific system resources attained appropriate status. They are similar to types of statistics gathered for Queueing Statistics. IPSS automatically collects these statistics as a by-product of the following IPSS Built-in procedures:

1. Wait Facility Status,
2. Wait Service Complete,
3. Wait I/O Complete,
4. Wait Semaphore,
5. Wait Access Path, and
6. Wait Signal.

The effect of these built-in procedures is to suspend the execution of a service until the identified condition is met. The statistics for each are the same. Only the WAIT FACILITY STATUS definitions are described in this section. Table C2.6-1 identifies all the IPSS Wait Statistics, Table C2.6-2 the means for their operation and Table C2.6-3 their definitions.

Table C.2.6-1 IPSS Wait Statistics

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition
Busy Period	Total number of busy periods	X		
	Number of zero-time busy periods	X		
	Percent zero-time busy periods	X		
	Total busy time	X		
	Percent busy	X		
	Mean length of a busy period	X		
	Mean length of a busy period (excluding zero-time periods)		X	EU-DP
	Standard deviation for non-zero time busy periods		X	EU-DP
	Maximum length of a busy period		X	EU-DP
	Minimum (non-zero) length of a busy period		X	EU-DP
	Length of the current busy period		X	EU-DP
Idle Period	Total number of idle periods	X		
	Number of zero-time idle periods	X		
	Percent zero-time idle periods	X		
	Total idle time	X		
	Percent idle	X		
	Mean length of an idle period	X		
	Mean length of an idle period (excluding zero-time periods)		X	EU-DP
	Standard deviation for non-zero time idle periods		X	EU-DP
	Maximum length of an idle period		X	EU-DP
	Minimum (non-zero) length of an idle period		X	EU-DP
	Length of the current idle period		X	EU-DP
Wait Chain Length Characteristics	Mean length	X		
	Maximum length	X		
	Current length	X		
	Standard deviation of mean length	X		

Table C.2.6-1 IPSS Wait Statistics (Continued)

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition
Wait Chain Entry Characteristics	Total number of entries	X		
	Total number of completed entries	X		
	Number of zero-time entries	X		
	Percent zero-time entries	X		
	Mean time in chain	X		
	Mean time in chain (excluding zero-time entries)	X		
	Standard deviation of mean time non-zero-time entries	X		
	Maximum time in chain		X	EU-DP
	Minimum (non-zero) time in chain		X	EU-DP

Disposition codes used in this table are:

EU-DP: Examined, found useful - defined, not implemented but useful in paper model.

Table C.2.6-2 IPSS Wait Statistics - Required Model Statements

Variable Type*	Description of the Variable	Method of Data Collection
WCNE	Current number of Concurrent wait chain elements	
WMXNCE	Maximum number of current concurrent wait chain elements	
WSNCE	Sum of number of concurrent wait chain elements	
WSNCES	Sum of (number of concurrent wait chain elements)**2	
WSWCE	Total number of wait chain elements	
WSZWCE	Total number of zero-time wait chain elements	
WSTT	Sum of time per transaction	All statistics are maintained by the Wait statements
WSTTS	Sum of (time per trans)**2	
WCTI	Cumulative Time integral	
WTWFSS	Time of last wait start or wait end	
WSBP	Total number of busy periods	
WSBPL	Sum of length of busy periods	
WSBPLS	Sum of (length of busy periods)**2	
WSIP	Number of idle periods	
WSIPL	Sum of length of idle periods	
WSIPLS	Sum of (length of idle periods)**2	
WTBIP	Time of last Busy or Idle period start	

Table C.2.6-2 IPSS Wait Statistics - Required Model Statements (Continued)

Variable Type*	Description of the Variable	Method of Data Collection
WSZBP	Number of zerotime busy periods	
WSZDP	Number of zerotime idle periods	
WMNBPL	Minimum non-zero busy period length	
WMXBPL	Maximum busy period length	
WCBPL	Length of current busy period	All statistics are maintained by the Wait statement
WMNIPL	Minimum non-zero idle period length	
WMXIPL	Maximum idle period length	
WCIPL	Length of current idle period	
WMXTT	Maximum transit time per transaction	
WMNTT	Minimum non-zero transit time per transaction	

*All are stored in the statistics area associated with the designated facility.

Table C.2.6-3 IPSS Wait Statistics - Definitions

Statistics Type	Specific Statistic	Definitions
Busy Period	Total number of busy periods	WSBP
	Number of zerotime busy periods	WSZBP
	%zero time busy periods	(WSZBP/WSBP)*100
	Total busy time	WSBPL
	%busy	(WSBPL/CLOCK3)*100
	Mean length of busy period	(WSBPL/WSBP)
	Standard deviation of mean length	((WSBP*WSBPLS-WSBPL**2) /(WSBP*(WSBP-1)))**0.5
	Mean length of non-zero time busy periods	(WSBPL/(WSBP-WSZBP)
	Standard deviation of non-zero time busy periods	(((WSBP-WSZBP)*WSBPLS-WSBPL**2)/((WSBP-WSZBP) *(WSBP-WSZBP-1)))**0.5
	Maximum length of a busy period	WMXBPL
	Minimum non-zero length	WMNBPL
	Length of current busy period	WCBPL
Idle Period	Total number of idle periods	WSIP
	Number of zerotime idle periods	WSZIP
	%Zerotime idle periods	(WSZIP/WSIP)*100
	Total idle time	WSIPL
	%Idle	(WSIPL/CLOCK3)*100
	Mean length of idle period	(WSIPL/WSIP)
	Standard deviation of mean length	((WSIP*WSIPLS-WSIPL**2) /(WSIP*(WSIP-1)))**0.5
	Mean length of non-zero time idle periods	(WSIPL/(WSIP-WSZIP))

Table C.2.6-3 IPSS Wait Statistics - Definitions (Continued)

Statistics Type	Specific Statistic	Definitions
	Standard deviation of non-zero time idle periods	(((WSIP-WSZIP)*WSIPLS-WSIPL**2) /((WSIP-WSZIP)*(WSIP-WSZIP-1)))**0.5
	Maximum length of an idle period	WMXIPL
	Minimum non-zero length	WXNIPL
	Length of current Idle period	WCIPL
Chain Length	Mean length	(WSNCE/WSWCE)
	Maximum length	WMXNCE
	Current length	WCNCE
	Standard deviation of mean length	((WSWCE*WSNCES-WSNCE**2) /(WSWCE*(WSWCE-1)))**0.5
Chain	Total number of wait chain entries	WSWCE
	Total number of completed chain entries	WSCCE = WSWCE - WCNCE
	Number of zero-time entries	WSZWCE
	%zero-time chain entries	(WSZWCE/WSWCE)*100
Transit Time	Mean transit time per transaction	(WSTT/WSCCE)*100
	Mean non-zero time per transaction	(WSTT/(WSCCE-WSZWCF))
	Standard deviation of mean transaction time	((WSCCE*WSTTS-WSTT**2) /(WSCCE*(WSCCE-1)))**0.5
	Standard deviation of mean non-zero-time transactions	(((WSCCE-WSZWCE)*WSTTS-WSTT**2)/((WSCCE-WSZWCE) *(WSCCE-WSZWCE-1)))**0.5
	Maximum transit time	QMXTT
	Minimum non-zero transit time	WMNTT

C.2.7 IPSS Service Related Statistics

IPSS Exogenous and Endogenous services are modeler-defined model components representing information system processes. These processes could be manual or automated. IPSS provides the modeler with over 100 statements in addition to the use of all of ANSI Fortran to define these services. In the current IPSS, services are treated the same as other acquirable resources for statistic gathering purposes.

In the proposal a number of additional statistics specific to services were proposed. The purpose of these new statistics is to characterize service performance in a more meaningful manner. This is accomplished in the following two ways:

A. Modeler Explicit Statistics

Three new sets of IPSS statements permit the modeler to divide overall service time into three segments;

1. Queue Time - time which a request for service spends in a service maintained queue waiting for its service to commence,
2. Acquire Time - time spent by the service acquiring the resources needed to service a request, and
3. Service Time - time spent by the service servicing the request.

These statistics are generated only when explicitly requested by the modeler through the employment respectively of the following IPSS statement pairs:

1. MARK QUEUE START - MARK QUEUE END
2. MARK ACQUIRE START - MARK ACQUIRE END
3. MARK SERVICE START - MARK SERVICE END

B. Modeler Implicit Statistics

The purpose of these statistics is to allocate a service's time to subordinate-services. The IPSS translator will identify

all Endogenous services (modeler defined or IPSS Built-in) requested by a modeler defined service. Statistics on subservice performance will be gathered and displayed with the calling service. These statistics show total service time for the invoked service with respect to the invocations by the requestor service and total time. The requestor service was delayed wait for the requested service to complete processing. These statistics are automatically generated by the REQUEST SERVICE - WAIT SERVICE statements.

Table C.2.7-1 identifies the specific statistics generated. Tables C.2.7-2 and C.2.7-3 specify the IPSS mechanisms used in their generation.

Table C.2.7-1 IPSS Service Related Statistics

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition*
Execution	Total number of executions		X	EU-DP
	Percent zero-time		X	EU-DP
	Mean execution time		X	EU-DP
	Mean non-zero execution time		X	EU-DP
Queue Time	Total number of periods		X	EU-DP
	Percent zero-time		X	EU-DP
	Mean period length		X	EU-DP
	Mean non-zero period length		X	EU-DP
	Mean percent of execution time		X	EU-DP
	Mean non-zero percent of execution time		X	EU-DP
Acquire Time	Total number of periods		X	EU-DP
	Percent zero-time		X	EU-DP
	Mean period length		X	EU-DP
	Mean non-zero period length		X	EU-DP
	Mean percent of execution time		X	EU-DP
	Mean non-zero percent of execution time		X	EU-DP
Service Time	Total number of periods		X	EU-DP
	Percent zero-time		X	EU-DP
	Mean period length		X	EU-DP
	Mean non-zero period length		X	EU-DP
	Mean percent of execution time		X	EU-DP
	Mean non-zero percent of execution time		X	EU-DP
Service by Service (For each Service invoked by a parent service)	Total number of invocations		X	EU-DP
	Percent zero-time invocations		X	EU-DP
	Mean time per invocation		X	EU-DP
	Mean time per invocation (excluding zero-time invocations)		X	EU-DP
	Total number of Wait Service Completes		X	EU-DP

Table C.2.7-1 IPSS Service Related Statistics (Continued)

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition*
	Percent zero-time Waits		X	EU-DP
	Mean time per Wait		X	EU-DP
	Mean time per Wait (excluding zero-time Waits)		X	EU-DP

*Disposition codes used in this table are:

EU-DP: Examined, found useful - defined, but not implemented

Table C.2.7-2 IPSS Service Related Statistics - Required Model Statistics

Variable Type	Description of the Variable	Method of Data Collection
The following variables identify storage locations in the fixed statistics area associated with each modeler defined Exogenous and Endogenous Service		
SNT	Total number of executions	Updated by REQUEST SERVICE Statement
SNZT	Total number of zero-time executions	Updated by WAIT RETURN* statement
SCUMT	Cumulative execution time	Update by TERMINATE* statement
SNQ	Number of queue periods	Updated by MARK QUEUE START statement
SNZQ	Number of zero-time queue periods	Updated by MARK QUEUE END, MARK ACQUIRE START, MARK SERVICE START or TERMINATE statements
SCUMQ	Cumulative queue period time	
SNAC	Number of acquire periods	Updated by MARK ACQUIRE START statement
SNZAC	Number of zero time acquire periods	Updated by MARK ACQUIRE END, MARK SERVICE START or TERMINATE statements
SCUMAC	Cumulative acquire period time	
SNS	Number of service periods	Updated by MARK SERVICE START statement
SNZS	Number of zero time periods	Updated by MARK SERVICE END or TERMINATE statements
SCUMS	Cumulative service period time	
The following variables identify storage locations in the variable part of the statistics area. These are replaced for each subordinate service requested by a service. (Note: intermediate statistics are kept in the transactions created for a requested service until the transaction is terminated.)		
$SSNI_A$	Total number of invocations of Service A	Updated by REQUEST SERVICE statement

Table C.2.7-2 IPSS Service Related Statistics - Required Model Statistics (Continued)

Variable Type	Description of the Variable	Method of Data Collection
$SSNZI_A$	Total number of zero-time invocations of Service A	Update by TERMINATE statement
$SSTMI_A$	Total execution time	Update by TERMINATE statement
$SSNW_A$	Total number of Wait Service	Update by WAIT SERVICE Complete statement
$SSNZW_A$	Total number of zerotime	Updated by WAIT RETURN statement
$SSTWS_A$	Total Wait service time	Updated by WAIT RETURN statement

*These statements are internal to IPSS and not available to the modeler.

Table C.2.7-3 IPSS Service Related Statistics - Definitions

Statistics Type	Specific Statistic	Definitions
Execution	Total number of executions	SNT
	Percent zero-time	(SNZT/SNT)*100
	Mean execution time	SAVT = (SCUMT/SNT)
	Mean non-zero time execution time	SAVTX = (SCUMT/(SNT - SNZT))
Queue Time	Total number of queue periods	SNQ
	Percent zero-time	(SNZQ/SNQ)*100
	Mean period length	SAVQ = (SCUMQ/SNQ)
	Mean non-zero time period length	SAVZQ = (SCUMQ/(SNQ - SNZQ))
	Mean percent of execution time	SAVQ/SAVT
	Mean non-zero % of non-zero execution time	SAVQZ/SAVTX
Acquire Time	Total number	SNAC
	Percent zero-time	(SNZAC/SNAC)*100
	Mean length	SAVAC = (SCNMAC/SNAC)
	Mean non-zero length	SAVACX = (SCUMAC/(SNAC - SNZAC))
	Mean % of execution time	(SAVAC/SAVT)*100
	Mean non-zero % of non-zero execution time	(SAVACX/SAVTX)*100
Service Time	Total number	SNS
	Percent zero-time	(SNZS/SNS)*100
	Mean length	SAVS = (SCUMS/SNS)

Table C.2.7-3 IPSS Service Related Statistics - Definitions (Continued)

Statistics Type	Specific Statistic	Definitions
	Mean non-zero length	SAVSX = (SCUMS/(SNS - SNZS))
	Mean % execution time	(SAVS/SAVT)*100
	Mean non-zero % of non-zero execution time	(SAVSX/SAVTX)*100
Subservice Behavior for Requestor Service	Total number of invocations	$SSNI_A$
	Percent zero-time	$(SSNZI_A/SSNI_A)*100$
	Mean time per invocation	$(SSTMI_A/SSNI_A)$
	Mean time per non-zero time execution	$(SSTMI_A/(SSNI_A - SSNZI_A))$
	Total number of WAIT SERVICE COMPLETES	$SSNW_A$
	Percent zero time Waits	$(SSNZW_A/SSNW_A)*100$
	Mean time per Wait	$(SSTW_A/SSNW_A)$
	Mean time per non-zero time Wait	$(SSTW_A/(SSNW_A - SSNZW_A))$

C.2.8 IPSS Task and Activity Statistics

These statistics provide insight into the service activity subordinate to a specific task, where a Task is a modeler defined facility activated as part of the execution of a Exogenous and Endogenous service. A Task is assumed to be comprised of a linear combination of one or more modeler defined activities which are identified as part of the Task declaration. A modeler defines an activity by associating an activity name with an endogenous service. Statistics are gathered for a Task by gathering statistics for its activities. This occurs in the following two step process:

1. The Task is explicitly activated via the INITIATE TASK statement.
2. Statistics are automatically gathered for each Endogenous service which
    a. is subordinate to the inservice containing the INITIATE TASK statement, and
    b. has been identified as an activity associated with the Task.

The statistics gathering continues until the TERMINATE TASK statement is executed.

Table C.2.8-1 identifies the statistics to be gathered. However, the actual collection mechanism to be used within an IPSS model has not yet been defined.

Table C.2.8-1 IPSS Task and Activity Statistics

Statistic Type	Specific Statistics	Currently Available	Proposed Extension	Disposition*
By Task -- General	Total number of tasks initiated		X	EU-IM
	Total number of tasks completed		X	EU-IM
	Total number of tasks still in execution		X	EU-IM
	Mean time per task initiation		X	EU-IM
By Activity -- Within Task -- Task Initiation Statistics	Total number of invocations by task		X	EU-IM
	Mean total activity processing		X	EU-IM
	Mean total queue time per task initiation		X	EU-DP
	Percent queue time of processing time		X	EU-DP
	Mean total activity acquire time per task initiation		X	EU-DP
	Percent acquire time of processing time		X	EU-DP
	Mean total activity service time per task initiation		X	EU-DP
	Percent service time of processing time		X	EU-DP
Activity Invoke Statistics	Mean processing time per invocation by task		X	EU-IM
	Mean activity queue time per invocation by task		X	EU-DP
	Percent queue time of processing time per invocation		X	EU-DP
	Mean activity acquire time per invocation by task		X	EU-DP
	Percent acquire time of processing time per invocation		X	EU-DP
	Mean activity service time per invocation by task		X	EU-DP
	Percent service time of processing time by task		X	EU-DP

*Disposition codes used in this table have the following meanings:

EU-IM:

EU-DP:

C3. IPSS EXTENSIONS TO CHARACTERIZE COMPUTER NETWORKS

The standard IPSS and the IPSS/DBS have been designed to operate as independent simulators. However, an integrated model capability is required in order to characterize a complete system. In the IPSS this capability was achieved by permitting the concurrent execution of two asynchronous simulation processes which have the ability to communicate. Once this cability was defined, it was possible to extend the mechanism to accommodate any number of asynchronous processes.

The purpose of this section is to describe the mechanism to be used which will permit the modeling of information processing system networks. For this discussion a mode is assumed to be an individual model containing IPSS components:

Component	Number Permitted
Request Streat	0 or 1
System Resources	0 or 1*
Storage Structure	0 or 1**
Data Base Access	0 or 1*
Data Structure	0 or 1***

*Either the System Resources or Data Base Access Component must be present. Both can also be present at each mode.

**Storage Structure Component can be present only when the System Resources Component is also present.

***Data Structure Component can be present only when the Data Base Access Component is also present.

A network is comprised of multiple nodes which communicate by initiating requests for services to another node. These services requests are treated the same as exogenous requests generated by a node's request stream. To establish a network the modeler defines a multiple facility pool - which is described in the following subsection.

C3.1 Multiple Facility Pools

Currently, in order to simulate a computer network, all nodes of the network have to be combined together to form a single facility pool consisting of at most one of each of the following components: system resource component, storage structure component, request stream component and model director component. Since the above approach does not facilitate independent testing of each node or evaluating the interaction between nodes, a modeling structure which allows multiple facility pools is needed. The basic structure of the IPSS nucleus which directs model execution is given in Figure C3.1-1a. In order to consider multiple facility pools, this structure will have to be modified to that shown in Figure C3.1-1b. In the multiple facility pool configuration, the nucleus consists of three drivers:

1. Model Driver
   The function of the model driver is the same as in the case of a single facility pool. It's basic function is to start and terminate the simulation drivers.
2. Global Simulation Driver
   The global simulation driver controls the FEQ (Future Events Queue) and the simulation clock. All future events generated by the different facility pools are combined together into a single FEQ so as to provide correct simulated time synchronization between the facility pools. The global simulation driver has a priority list that is used to determine which local simulation driver has control.
3. Local Simulation Driver
   Each facility pool has a local simulation driver which controls its corresponding CEQ (Current Event Queue). Once a local simulation driver is given control, it retains control until either (i) a transaction has to be placed in a different CEQ or (ii) a transaction has to be placed in the FEQ. In these cases, the local driver

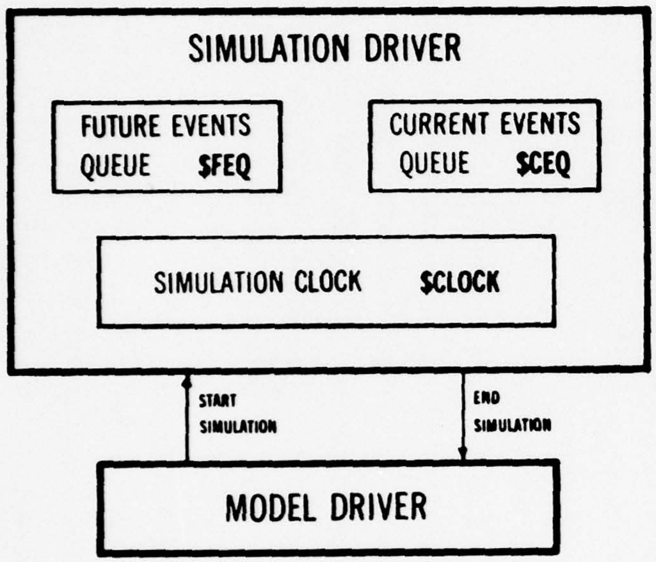


Figure C3.1-1a Basic Structure of IPSS Simulation Nucleus

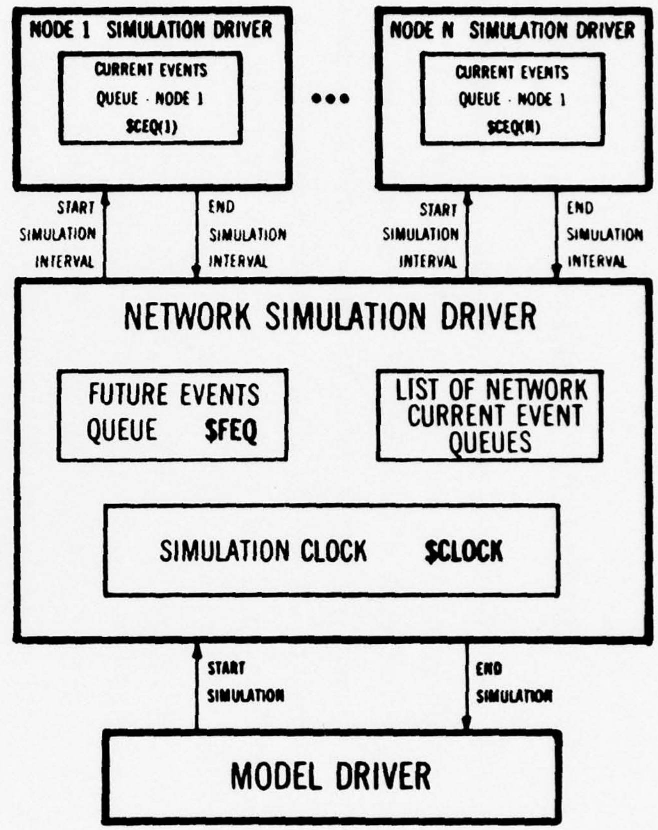


Figure C3.1-1b Structure of Model Nucleus to Support Networking

returns control to the global driver which has the responsibility for placing the transaction into the FEQ or calling the appropriate local nucleus to insert a transaction into its CEQ. If placement is in a CEQ with higher priority than the currently active one, the interrupt flag is set. The currently active nucleus again resumes processing its active transaction until its processing is suspended. If the interrupt flag is set, control is given to the local driver with the higher priority via the global nucleus, otherwide the current driver continues with its next transaction. The process continues until all the CEQ's are empty at which time the clock is advanced to the next most imminent departure time in the FEQ. One or more transactions with this time are then moved to their corresponding CEQ and control is given to the nucleus with highest priority and the process begins a new.

C3.2 Changes to IPSS Simulation Nucleus

In order to accommodate the multiple facility pool structure, a number of changes had to be made to the IPSS nucleus and system subroutines.

1. The IPSS definitional tables (e.g., $TOC, $LBL, $DIR, $DEF) of the individual facility pools have to be combined together to form a single set of tables.
2. To avoid naming conflicts between facility pools, a global interface routine is needed to invoke the currently implemented local interface routines.
3. The system subroutines (e.g., EQUATE, INVOKE) that handle intra-facility pool communications have to be modified to handle inter-facility pool communications.

Detailed programming changes to IPSS are given below.

Changes to the IPSS Fortran System Subroutines to Handle Multiple Facility Pools

(A) The following procedures are to be modified:

1. $FDEF

   This procedure has to be modified so that it will accept base pointers to the definitional tables as parameters. A procedure for searching the System Resources and Data Base Access $DEF tables through $TOC, $LBL and $DIR has to be added to this procedure.

2. $EQUAT

   This procedure has to be modifed so that it will accept definitional table addresses and bases for the equated components as input parameters.

3. $CREAT

   This procedure has to be modified so that both the Facility's Pool ID and Driver ID are stored in the created transaction.

4. $GENER, $GADR, $SYSEX2, $SEEK, $EXINT

   All calls to the interface routine are to be modified.

5. $LINK, $UNLINK, $INSRT, $REMOV

   All these procedures have to be modified to accept multiple CEQ's.

6. $DRIVR

   This procedure has to be modified to handle multiple CEQ's. The driver should automatically interchange base for the definitional table when a new CEQ is selected. All calls to interface routines should be modified. Also, all calls to initialization routines have to be modified so that all Facility Pool/Driver-relative components are initialized.

7. $DVINT, $ABORT

   Parameter lists are to be deleted from these procedures and appropriate COMMON's should be added.

8. All statistical routines are to be modified so that separate statistics will be printed for each facility pool.
9. All procedures that use the Create Data Set ($CDS) arrays have to be changed.
10. All procedures that use absolute addresses for accessing the definitional table have to be changed so that they use relative addresses in the base table.

(B) The following procedures to be added.

1. $BLOC
   When the Facility Pool ID and Driver ID are given, this procedure will search through the base table to identify the bases to be used for the local simulation driver.
2. $SUPEQ
   This procedure handles all inter-facility pool equates and makes use of $EQUAT to resolve all intra-facility pool equates.
3. $INIT
   This procedure initializes all endo and exo services with the facility pool and driver ID's. The endo and exo service definitions are found by searching through $TOC, $LBL and $DIR.
4. $INVEX
   This procedure handles the invoking of an Exo service in another facility pool. Basically, this procedure facilitates inter-facility pool communications.

(C) The following Fortran tables and statements are to be generated by the language parsers.

1. CEQ Table
   This table will be used by the flobal simulation driver to determine the priority of the CEQ's.

$PRTY	$FPID	$DRIVR	$CBEG	$CEND
.	.	.	.	.
.	.	.	.	.
.	.	.	.	.

$PRTY: Priority of the CEQ

$DPIF: Facility Pool ID

$DRIVR: type of driver

1. if SR, RS or SS component driver
2. if DBA or DT component driver

$CBEG: beginning addresses of CEQ's (initialized to zero's)

$CEND: ending addresses of CEQ's (initialized to zero's)

2. BASE Table

This table stores the bases used for addressing the definitional tables.

$FPID	$CMPNT	$$TOC	$$LBL	$$DIR	$$DEF	$CDS	$NEVNT
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.

$FPID: Facility Pool ID

$CMPNT: Component ID

type = 1 if SR component

type = 2 if RS component

type = 3 if SS component

type = 4 if DBA component

type = 5 if DT component

$$TOC: list of bases for $TØC

$$ LBL: list of bases for $LBL

$$DIR: list of bases for $dir

$$DEF: list of bases for $DEF

$CDS: list of bases for create data set array

$NEVNT: number of events that has occurred in the associated facility pool.

3. The following interface routine will be generated for each (facility pool, driver) pair.

```
      SUBROUTINE $xxxxx (TYPE, SYSIND, INDX1, INDX2,*)
      INTEGER TYPE, SYSIND, INDX1, INDX2
      GO TO (1ØØ, 2ØØ, 3ØØ, 4ØØ), TYPE
   5Ø RETURN 1
  1ØØ CALL $CFPR (SYSIND, &5Ø)
      RETURN
  2ØØ CALL SYSER (INDX2, INDX1, &5Ø)
      RETURN
  3ØØ CALL $DBPR (SYSIND, &5Ø)
      RETURN
  4ØØ CALL $EXPR (SYSIND, &5Ø)
      RETURN
      END
```

Where xxxxx is a unique name generated for each (facility pool, driver) pair.

4. The following global interface routine is generated for the model pool.

```
      SUBROUTINE $SUPER (FPID, DRIV, TYPE, SYSIND, INDX1, XINDX2,*)
      INTEGER FPID, DRIV, TYPE, SYSIND
      INTEGER INDX1, INDX2, I
      GO TO 9999
1     CALL $xxxxx (TYPE, SYSIND INDX1, INDX2 $1ØØØØ)
      RETURN
      .
      .
      .
```

```
9999  I = FPID *2 + DRV-2
      GO TO (1, 2, ... n), I
10000 RETURN 1
      END
```

Where n is equal to the number of (facility pool, driver) pairs.